University of Alberta

AN INTELLIGENT 3-D OPEN PIT DESIGN AND OPTIMIZATION USING MACHINE LEARNING – ADAPTIVE LOGIC NETWORKS AND NEURO-GENETIC ALGORITHMS

by
Eric Asa ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy

in

Mining Engineering

Department of Civil and Environmental Engineering

Edmonton Alberta

Spring 2002

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-68537-3

Canada

University of Alberta

Library Release Form

Name of Author : **Eric Asa**

Title of Thesis : An Intelligent 3-D Open Pit Design and Optimization Using Machine Learning – Adaptive Logic Network and Neuro-Genetic Algorithms

Degree: Doctor of Philosophy

Year this Degree Granted: 2002

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as here in before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

132 Row House, Michener Park, Edmonton,

AB, T6H 4M4, Canada

Dated: April 16, 2002

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommended to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled "*An Intelligent 3-D Open Pit Design and Optimization Using Machine Learning – Adaptive Logic Network and Neuro-Genetic Algorithms*" submitted by Eric Asa in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Mining Engineering.

Dr. Samuel Frimpong (Supervisor)

Dr. Jozef Szymanski

Dr. Steve Zou (External Examiner)

Dr. Witold Pedrycz

A Robinson Fayek

Dr. Aminah Robinson Fayek

Dr. Ergun Kuru

Date: February 28, 2002.

ABSTRACT

The stochastic nature of the parameters involved in the design and optimization of a three-dimensional (3-D) open pit mine was not taken into consideration in most of the earlier work and the resulting optimized pit may be sub-optimal. The only recent significant contribution is the use of a back-propagation neural network algorithm to optimize a two-dimensional pit. This thesis is therefore aimed at using machine learning algorithms to design and optimize a 3-D open pit mine.

After a detailed initial literature review of the design and optimization of open pit mines, geostatistics, intelligent algorithms and software are used to develop 3- D block (gold and coal) models. The resulting intelligent neurogenetic block predictors are employed in predicting the values of the gold and coal ore values. A slope model is also developed and used together with the block model in the optimization of the pit. A combination of simulated annealing and neurogenetic optimizer (SA-NGO) is then used to optimize the open pit gold mine. The intelligent neurogenetic pit optimizer and Lerchs-Grossmann's 3-D graph theory algorithm are used to run 30 experiments. The results are compared.

The block (gold and coal) model results from several algorithms including adaptive logic network were inaccurate. The results of the stochastic approach – generalized regression neural networks for both the gold and coal block ore were accurate. It is therefore evident that a blind search or black-box approach to intelligent computational modelling may lead to faulty results. The underlying phenomenon has to be taken into consideration in the modelling process. The optimum pit values of the L-G and SA-NGO algorithms were $29.8 million and $27.5211 million, respectively. The major contributions of this work are the development of a formal procedure for geostatistical ore-body modeling, an intelligent 3-D block predictor and an intelligent 3-D open pit optimizer. The resulting block predictor and open pit optimizer can be downloaded into the onboard computer of the excavation equipment and used for just-in-time operational decisions.

ACKNOWLEDGEMENT

I wish to express my sincere appreciation to my advisor, Dr. Samuel Frimpong, without whose moral and financial support and guidance this work would not have been completed. My gratitude is also extended to the members of my committee, namely Dr. Jozef Szymanski, Dr. Steve Zou (External Examiner, Dalhousie University), Dr. Witold Pedrycz, Dr. Aminah Robinson Fayek and Dr. Ergun Kuru, for their comments, time and effort. I am grateful to the Department of Civil and Environmental Engineering for the tuition scholarships, sessional appointments and teaching/graduate assistantships, which accorded me considerable experience on one part and supplemented my income on the other hand. I am equally indebted to Dr. Bill Armstrong of Dendronic, BioComp and Gensym Intelligent Real Time Systems for giving me free copies of their software for use in the course of this work.

Finally I reserve the rest of my acknowledgements for my wife, Sophia Acquah Asa, who worked tirelessly to support the family and to my children, Godfried Samuel Asa Attuah, Gladys Christiana Adubea Asa, Daniel Asa and Emelia Gyasiwah Asa, who had to endure many dull nights.

TABLE OF CONTENTS

ABSTRACT

ACKNOWLEDGEMENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF NOMENCLATURE

CHAPTER 1: INTRODUCTION

1.1 Background 1

1.2 Statement of the Problem 3

1.3 Objectives of the Study 6

1.4 Scope and Limitations of the Study 7

1.5 Research Methodology 8

1.6 Contributions and Industrial Significance of the Study 11

1.7 Report Structure 13

CHAPTER 2: DETAILED LITERATURE REVIEW

2.1 Geomechanical Mine Design 16

2.2 Geometrical Mine Design 23

2.2.1 Bench Height Design 24

2.2.2 Bench Width Design 25

2.2.3 Width of Safety Berm 26

2.2.4 Imposition of Slope Walls....27

2.2.5 Imposition of Haul Road....31

2.3 Computer-Aided Mine Design....32

2.3.1 Geostatistical Modelling in Computer-Aided Mine Design....34

2.4 Open Pit Optimization Algorithms....36

2.4.1 The Manual Approach to Open Pit Design and Optimization....37

2.4.2 The Moving Cone Algorithm....39

2.4.3 Lerchs-Grossmann's Optimum Open Pit Mine Design....42

2.4.4 The Dynamic Programming Algorithm by Koenigsberg....43

2.4.5 3D Dynamic Programming (Wilke and Wright)....45

2.4.6 Dynamic Path Level Pit Design and Optimization....46

2.4.7 Machine Learning Algorithms and Open Pit Mine design and Optimization....46

2.4.8 Neural Network ((MCS/MFNN) Algorithm for Open Pit Optimization......47

2.5 Overview of Machine Learning....48

2.5.1 Neural Network....49

2.5.2 Multi-layer Feed-forward Neural Network....52

2.5.3 Hopfield Network....54

2.5.4 Kohonen Network....55

2.5.5 Conclusion 56

CHAPTER 3: THEORY AND PHILOSOPHY OF OPEN PIT DESIGN AND OPTIMIZATION

3.1 Design Paradigms and Optimality Criteria……57

3.1.1 Design Evaluation……60

3.1.2 Design Uncertainty……62

3.1.3 Interaction of Design characteristics……64

3.1.4 Optimization and Optimality Criteria……65

3.2 Computer Modelling of Mineral Deposits……67

3.2.1 Block Modelling……68

3.2.2 Block Model for Open Pit Design……71

3.3 The Economic Value of a Block……73

3.4 Rock Slope Design……74

3.4.1 Slope Design……74

3.4.2 Uncertainty Analysis……75

3.5 Optimization of Engineering Systems……76

3.5.1 Combinatorial Optimization……81

3.5.2 Global Optimization……81

3.5.3 Stochastic (Probabilistic) Global Optimization……83

3.6 Conclusions……84

CHAPTER 4: MATHEMATICAL MODELLING USING MACHINE LEARNING

4.1 Design Paradigms……86

4.1.1 Design Paradigms for Machine Learning Algorithms....87

4.1.2 Hypothesis Testing....88

4.2 Mathematical Modelling by Adaptive Logic Networks....90

4.2.1 Perceptron Network....91

4.2.2 Adaptive Logic Network (ALN)....92

4.2.3 Network Architecture of ALN....95

4.3 Mathematical Modeling by Continuous Adaptive Time Networks....97

4.4 Mathematical Modeling by Generalized Regression Networks....101

4.5 Decision Analysis in Design and Optimization....102

4.6 Simulated Annealing....107

4.6.1 Boltzmann Annealing....109

4.6.2 Simulated Quenching....109

4.6.3 Fast Annealing....110

4.7 Genetic Algorithms....110

4.8 Neuro-Genetic Algorithms....112

4.9 Conclusions....112

CHAPTER 5: COMPUTER MODELLING

5.1 Introduction....114

5.1.1 Transformation of Mathematical Models into Computer Models....114

5.2 Adaptive Logic Network Computational Algorithm....115

5.3 Computerization of Three-Dimensional (3-D) Optimization Algorithm....119

5.4 Search Procedure and Situation Calculus of the Intelligent Agent....122

5.5 Tracking Location..............................123

5.6 Stochastic Optimization Methods..............................123

5.6.1 Random Search Algorithms..............................124

5.6.2 Heuristic Repair..............................124

5.6.3 Random Incremental Improvement Algorithm..............................125

5.6.4 Monte Carlo Simulation..............................125

5.7 Logic Flow Network for Simulated Annealing Algorithm..............................126

5.8 Pseudo-Code Development for Simulated Annealing..............................128

5.9 Simplified Computer Program Development Process..............................129

5.10 Object-Oriented Design of the Optimizer..............................130

5.11 Requirements Analysis..............................130

5.12 Identification of Objects for Open Pit Optimization Problem..............................131

5.13 Neuro-Genetic Algorithm..............................132

5.13.1 Using Genetic Algorithms to Design Neural Network Architectures......133

5.14 Computer Models for Computer-Aided Design (CAD)..............................134

5.15 Conclusion..............................134

CHAPTER 6: EXPERIMENTAL DESIGN AND EXPERIMENTATION

6.1 Introduction..............................135

6.1.1 Machine Learning Algorithms..............................135

6.2 Verification and Validation..............................136

6.3 Experimental Design..............................137

6.3.1 Experimental Design Framework and Model Correctness..............................139

6.3.2 Experimental Design for Optimized 3D Open Pit Mine........................141

6.3.3 Experimental Design for Optimized Pit Slopes.................................145

6.4 Block Modelling Experiments Using Geostatistics.............................146

6.4.1 Location...147

6.4.2 Description of Input Data..147

6.4.3 Geology and Background Information.......................................148

6.5 Block Modelling...149

6.5.1 Machine Learning (Adaptive Logic Network) Experiment.....................149

6.6 Grade and Block Modelling with Neuro-Genetic Algorithm....................152

6.7 Open Pit Slope Design Experiments with Neuro-genetic Algorithm..........154

6.8 Design of Optimization Experiments..155

6.8.1 Optimization by Simulated Annealing..156

6.8.2 Risk and Sensitivity Analysis and Optimization by Neurogenetic Algorithm...161

6.8.3 Optimization by Lerchs-Grossmann's Algorithm..............................161

6.8.4 Comparative/Statistical Analysis of Open Pit Optimization Algorithm......162

6.9 Conclusions...163

CHAPTER 7: DISCUSSION OF RESULTS

7.1 Resource Modelling and Evaluation...164

7.1.1 Exploration Data (Statistical) Analysis...165

7.1.2 Variogram Modelling..166

7.1.3 Model Correctness...169

7.1.4 Kriging....170
7.1.5 Simulation....171
7.1.6 Resources/Reserves Reporting....172
7.2 Adaptive Logic Network....173
7.2.1 A Detailed Example of ALN Training or Estimation....174
7.3 Neuro-Genetic Modelling of Block Values....178
7.3.1 Modelling of Gold Block Values....179
7.3.2 Results of Modelling of Coal Block Values....187
7.4 Significance of Prediction of Block Values Using the Neuro-Genetic Algorithm....197
7.5 Results of Open Pit Optimization by Simulated Annealing and Neuro-Genetic Algorithms....198
7.6 Sensitivity and Comparative Analysis of Results of Simulated - Neuro-Genetic and Lerchs-Grossmann Algorithms....203
7.7 Stripping Ratio Results....208
7.8 Graphical Representation of An Optimized Pit....212
7.9 Conclusions....213

CHAPTER 8 CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusions....214
8.2 Recommendations....217

LIST OF REFERENCES..219

APPENDIX 5.1 ADAPTIVE LOGIC NETWORK OBJECTS 230

APPENDIX 6.1 PART OF ROCK TYPE 1 DATA............................233

APPENDIX 6.2 PART OF ROCK TYPE 2 DATA 234

APPENDIX 6.3 PART OF BLOCK MODEL OF A GOLD MINE........236

APPENDIX 6.4 PART OF BLOCK MODEL OF A GOLD MINE 237

APPENDIX 6.5 D45BLK.C PROGRAM...239

APPENDIX 6.6 SIM3D.C PROGRAM..241

APPENDIX 7.1 RESULTS OF FIRST GOLD BLOCK MODEL ALN CREATED 259

APPENDIX 7.2 RESULTS OF SECOND GOLD BLOCK MODEL ALN CREATED 260

APPENDIX 7.3 ARCHITECTURE OF SECOND GOLD BLOCK MODEL ALN CREATED...261

APPENDIX 7.4 MULTILAYER FEED-FORWARD NEUROGENETIC (BACK-PROPAGATION) NETWORK..................264

APPENDIX 7.5 GENERALIZED REGRESSION NEUROGENETIC (OPTIMIZER) NETWORK.................................266

LIST OF TABLES

Table 3.1: Geometrical Modelling Techniques for Orebodies---------------------------- 67

Table 7.1: Summary Statistics of LG and SA-NGO--206

LIST OF FIGURES

Figure 2.1: Pit Slope Design Without a Ramp--27

Figure 2.2 : Pit Slope Design With a Ramp/Road--29

Figure 2.3: Block Diagram for the Floating Method------------------------------------ 40

Figure 2.4: The 12 Nearest Neighbours to Block b_{ijk} in the Backward Direction ----44

Figure 2.5: A Local and Global Maxima in a Typical Optimization Problem--------- 47

Figure 2.6: A Biological Neuron---49

Figure 2.7: McCulloch-Pitts Neuron Model---50

Figure 2.8: Mathematical Representation of Activation Functions---------------------51

Figure 2.9: Multilayer Feedforward Neural Network-------------------------------------52

Figure 2.10: Hopfield Network 2^{-1} --54

Figure 2.11: Kohonen Network---55

Figure 3.1: Generalized Design and Optimization Model--------------------------------57

Figure 3.2: Solution for a Typical Constrained Design------------------------------------58

Figure 3.3: The Domain of the Solution Space--59

Figure 3.4: The Optimization Process---60

Figure 3.5: A Series Combination of Design Factors-------------------------------------61

Figure 3.6: Design Characteristics Values with a Uniform Distribution----------------63

Figure 3.7: Design Characteristics Values with a Non-Uniform Distribution----------63

Figure 3.8: A Traditional Open Pit Design and Optimization Process------------------65

Figure 3.9: An Intelligent Open Pit Design and Optimization Process-----------------66

Figure 3.10: Sequence of Action in Geostatistical Modelling and Evaluation of
Ore Reserves--69

Figure 3.11: Large Rectangular Box to Encompass Area of Interest--------------------69

Figure 3.12: Large Box is Subdivided into Smaller Three-Dimensional Blocks--------69

Figure 3.13: Classification of Engineering Systems-------------------------------------76

Figure 3.14: Difficulties in Solving Nonlinear Optimization Problems----------------78

Figure 3.15: Multiple Model Approach to Optimization of Nonlinear Systems-------79

Figure 3.16: A System Within a Universe---80

Figure 3.17: Local and Global Optima for Constrained Optimization Problems------ 82

Figure 3.18: Local and Global Optima for Unconstrained Optimization Problems--- 83

Figure 3.19: Stochastic Global Optimization---84

Figure 4.1: Relationship Between Sample and Universal Space------------------------87

Figure 4.2: True and Chosen Hypotheses---89

Figure 4.3: Representation of a Function by ALN (After Armstrong et al., 1995)-----94

Figure 4.4: Sequential Motion of a 3-D Intelligent Agent--------------------------------105

Figure 4.5: Sequential Motion of a 3-D Intelligent Agent Through a Typical Open Pit--106

Figure 4.6: Process of Breakdown of a Generation into Selection and Recombination Phases and Assignment of Strings into Slots--------------111

Figure 5.1: Transformation of Mathematical Models into Computer Models---------114

Figure 5.2: Flowchart for Transforming ALN Equations into Code/Software---------115

Figure 5.3: Computational Architecture for the Adaptive Logic Network--------------117

Figure 5.4: An Incremental 3-D Open Pit--119

Figure 5.5: A Section Showing the Incremental Movement of the 3-D Intelligent Agent---120

Figure 5.6: Incremental Optimization Algorithm--121

Figure 5.7: Location Tracking in An Intelligent Agent-----------------------------------123

Figure 5.8: Dynamics of Iterative Algorithms---124

Figure 5.9: Flow Diagram of a Simulated Annealing Algorithm-------------------------127

Figure 5.10: Pseudo-Code for Homogeneous Simulated Annealing-------------------128

Figure 5.11: Flow Diagram of the Object-Oriented Design Method-------------------129

Figure 5.12: Using Genetic Algorithms to Configure Network Problems(After NeuroGENESYS, 2001)---131

Figure 5.13: Using Genetic Algorithm to Design Neural Network Architecture--------132

Figure 6.1: Methodology for Designing Correct Experiments--------------------------140

Figure 6.2: Design and Optimization Experiments--------------------------------------141

Figure 6.3: Intelligent Block Design Paradigm (Gold Mine)---------------------------142

Figure 6.4: Flowchart for the Pit Optimization Process Using Simulated Annealing - Neurogenetic Optimizer ---------------------------------144

Figure 6.5: Sequence of Actions in Reserve Modelling and Evaluation-------------146

Figure 6.6: ALN Experimentation Processes--149

Figure 6.7: Open Pit Slope Design with Neuro-Genetic Algorithm-------------------154

Figure 6.8: Design of Optimization Experiments---------------------------------------155

Figure 6.9: Boltzmann Probability Distribution of Energy States According to Temperature (After Spinellis and Papadopoulos, 2000)--------------------160

Figure 6.10: Trajectory of a Simulated Annealing Experiment-------------------------160

Figure 6.11: Comparison of Optimization Resutls--162

Figure 7.1: Histogram of Rock Type 2--165

Figure 7.2: Cumulative Probability Plot of Rock Type 2--------------------------------166

Figure 7.3: Normal Scores Variogram in Horizontal Direction-------------------------167

Figure 7.4: Horizontal Variograms in the N22.5^{0}E, N67.5^{0}E, N112.5^{0}E and N157.5^{0}E Directions --167

Figure 7.5: Vertical Variograms in the N22.5^{0}E, N67.5^{0}E, N112.5^{0}E and N157.5^{0}E Directions --168

Figure 7.6: A Screen shot of the ALN for the Block Values---------------------------176

Figure 7.7: The Behaviour of the Error Function Durinh Training and Testing------177

Figure 7.8: A Typical Neuro-Genetic Modelling Process-------------------------------179

Figure 7.9: Portion of Genetic Population for Gold Block Model----------------------180

Figure 7.10: Generation of New Population (After Looney, 1997)--------------------181

Figure 7.11: Desired and Predicted Values From Back-Propagation Model-----------182

Figure 7.12: Graph of Experimental Versus Actual Results Using the Back-Propagation Neural Network--184

Figure 7.13: Desired Versus Predicted Values from GRNN-----------------------------185

Figure 7.14: Graph of Experimental Versus Actual Results Using GRNN-------------186

Figure 7.15: Predicted Versus Actual Block Values--------------------------------------187

Figure 7.16: Continuous Adaptive Time Neural Network Model for Coal Block Model --188

Figure 7.17: Desired Versus Predicted Values from CATNN – Coal Seam 1---------189

Figure 7.18: Desired Versus Predicted Values from CATNN – Coal Seam 2---------190

Figure 7.19: CATNN Experimental Results for Coal Block Model – Seam 1--------191

Figure 7.20: CATNN Experimental Results for Coal Block Model------------------192

Figure 7.21: Desired and Predicted Values from GRNN – Coal Seam 1-------------193

Figure 7.22: Desired and Predicted Values from GRNN – Coal Seam 2--------------194

Figure 7.23: GRNN Experimental Results for Coal Block Model - Seam 1----------195

Figure 7.24: GRNN Experimental Results for Coal Block Model - Seam 2----------196

Figure 7.25: Predictions from Neuro-Genetic Back-Propagation Neural Net--------199

Figure 7.26: Predictions from Neuro-Genetic General Regression Neural Net-------200

Figure 7.27: Graph of Desired Vs. Predicted Pit Values – BP------------------------201

Figure 7.28: Graph of Desired Vs. Predicted Pit Values – GRNN--------------------202

Figure 7.29: Open Pit Optimized Using the NGO--------------------------------------203

Figure 7.30: Graph of Lerchs-Grossmann and Neurogenetic Optimizer Pit Values--204

Figure 7.31: Graph of Alpha Values and Confidence Limits for LG and NGO Pits--207

Figure 7.32: Graph of Pit Values versus Stripping Ratio for Lerchs-Grossmann Algorithm--209

Figure 7.33: Graph of Pit Values versus Stripping Ratio for SA-NGO Algorithm--210

Figure 7.34: Graph of Pit Values versus Stripping Ratio for SA-NGO and L-G Algorithm--211

Figure 7.35: Three-Dimensional Representation of Three SA-NGO Pits------------212

LIST OF NOMENCLATURE

ξ = angle between the wedge - forming planes

ϕ = friction angle

ψ_i = plunge of the line of intersection

x_i = the presented input pattern; x_0 is –1

ρ = density of the material

δ = inclination of the upper ground surface

γ = unit weight of rock or soil material

β_1 = overall slope angle of the lower section

β_2 = overall slope angle of the upper section

η_{av} = cumulative efficiency factors

ψ_b = dip of the base of the columns

ϕ_b = friction angle of the base of the columns

Ψ_f = angle of slope face and d denotes unit depth

Ψ_f = dip of the slope face

β_L = overall slope angle of the pit according to Lerchs-Grossmann

φ_p = angle of intersection or the angle between the failure and horizontal planes of the open pit slope

Ψ_p = dip of the geologic layers (planes)

ϕ_p = friction angle along planes

ϕ_p = plane friction angle

Ψ_p = slope angle of failure plane

α_{r1} = geotechnically-determined slope angle of the lower section

α_{r2} = geotechnically-determined slope angle of the upper section

α_{ri} = batter angle or the angle the slope face makes with horizontal

γ_w = unit weight of water

Δx = column width

Δx_a = actual column width

Δxl = theoretical limiting column width or the column width at which toppling failure

γ_r = unit weight of rock

A = area of the failure plane

A_f = failure due to discontinuity; B_f = failure due to excessive water pressure

ψ_a and ψ_b = dips of planes a and b respectively

ϕ_a and ϕ_b = angles of friction on planes a and b respectively

b_1 = bench width of the lower set of benches

b_2 = bench width of the upper set of benches

B_h = bench height (assumed as being the same for all the benches)

b_i = distance between the safety berm and toe of the safety bench

b_{rw} = width of safety berm; c_r = clearance; TW = truck width

BSi in column (j-1, k-1) = back of side of i

c = cohesion

c_a and c_b = cohesive strengths of planes a and b respectively

CC_t = capital cost deductions.

DC = Direct costs or costs of mining that can be traced directly back to the block; e.g. drilling, blasting, loading and transportation costs.

DOR = radius of clean-up

g_j, Y_j, c_j = grade, tonnage and unit mining costs of the jth ore block

H = height of slope

h = total height of the wedge

$H_1, H_2, H_3, ..., H_N$ = all the nodes of the first hidden layer

h_a = height of haulage truck axle

I = Income or value of the recoverable and salable part of the block;

i = the required, risk-adjusted rate of return.

IC = Indirect costs or overall costs which cannot be allocated to individual blocks. Such costs are time-dependent; e.g. depreciation for machinery, etc.

M = number of inputs

M_{ijk} =the block column for a given block b_{ijk}

M_n = moment arm of force P_n

N_l = total number of blocks over entire property less non-mineable areas.

$O_1, O_2, O_3, ..., O_N$ = the nodes of the output layer occurs

OC_t = other allowable-tax-deductible operating costs

O_{ip} = activation value (network output) at output node o for pattern p on node I

$P_{BSi,j-1,k-1}$ = the pit value on a block in a column (j-1,k-1) to be selected as the optimum neighbour block, in the event the block $b_{SBS(Si), j-1, k-1}$ turns out to be incompatible to both b_{ijk} and block $b_{BSi,j,k}$.

P_{ijk} = the optimum value of the pit with the block b_{ijk} as the last block to be analyzed

P_{n-1} = the toppling force of the (n-1)th block

$P_{SBS(Si),j-1,k-1}$ = the pit value on the block which was the optimum neighbour block in the column (j-1,k-1), determined during the computation of $P_{si,j-1,k}$

$P_{Si,j-1,k}$ = the pit value of one of the nearest neighbour blocks in the column (j-1, k)

Q = maximum tonnage of recovered metal from a technically optimum pit

r = angle of repose of the berm or bench material

RCR = dumping radius

R_t = total mining royalties at time t and T_t = total mining taxes at time t.

SBSi in column (j,k-1) = side of back of side of i

Si in column (j-1,k) = side of i

ß = angle between the line of intersection and the horizontal in wedge failure

SSBSi in column (j,k-1) = side of side of back of side of I

t = number of roads within the n benches ($t < n$)

T = tons of ore in the pit

tanh(x) = tangent function

TC_t = total cost at time t and NCC_t = non-cash costs at time t.

Tx_t = income tax rate for the period t.

U = uplift from the water

V = disturbing force from water

V = given tons of ore mined from a technically optimum pit

W = weight of block

w_i = weight of the ith neuron

w_0 = the threshold value

W_{ho} = weight associated with the connection between node h of the hidden layer and node of the output layer

W_{jh} = weight associated with the connection between node j of the input layer and node h of hidden layer.

W_n = weight of the nth block

W_T = total number of mineable waste blocks over entire property.

$x_1, x_2, x_3, \ldots\ldots, x_N$ = all the nodes within the input layer

x_j = value of the input node

y = an output unit

y_n = height of the blocks/columns

Y_T = total number of mineable ore blocks over entire property.

Z = distance below horizontal surface

Z_w = depth of water in the tension crack

CHAPTER 1

INTRODUCTION

1.1 Background

The successful development, exploitation and closure of an open pit mine are very complex undertakings, extending over several years and with huge sums of capital investments and considerable risk and uncertainty. In view of the large initial capital outlays involved, mining engineers must carry out extensive mine design and optimization prior to mining, in order to establish the viability of the operation. Open pit optimization methods have drawn considerable attention in the past 30 years. It is going to be even more important as prices fall and environmental costs rise.

Modern society needs a supply of minerals for its growth and sustenance. Most of these minerals are obtained by means of surface and underground mining technology. Currently, surface mining accounts for a significant proportion of produced minerals. Surface mining operations generally have many advantages in terms of the sizes of production equipment; pre-production development period, ore recovery and labour requirements compared with underground. Surface mines are classified into open pit, strip, alluvial and in-situ mining methods [Hartman, 1987; Nilson, 1982]. Open pit mining layouts consist of concentric shells with near-ellipsoidal cross-sections. These shells decrease in size with increasing depth from the surface. Wall slopes, whose angles depend on the rock mechanics and geological characteristics of the ore body and host rocks, bound an open pit layout. The challenge to mine planning engineers is to plan, design and optimize the pit layouts to minimize waste removal, ensure safety and maximize the net value of the minerals in the pit.

The configuration of optimized pit layouts is one of the most important tasks in the overall open pit mine design process, which has to be determined right at the very beginning of mine planning. These layouts must continuously be readjusted throughout the life of the mine due to changing database of geology, ore grades and price [Wilke, 1990]. The optimized pit limits define the size and shape of mineable reserves and the associated waste materials to be excavated based on the technical, economic and safety constraints. They also provide information for evaluating the economic potential of a mineral deposit, and for project acquisition, financing, taxation, regulation and the formulation of long-, intermediate-, and short-range mine plans. Open pit limits are also used to determine the boundaries outside which surface structures, such as, processing plants and mine offices should be located in order to avoid interruption in the long-term mine plans.

Lerchs and Grossmann (1965) published one of the most important mathematical algorithms for optimizing open pit limits based on dynamic programming (2D) and graph theory (3D). Many algorithms have been developed to solve the problems associated with the Lerchs-Grossmann's algorithm. However, these problems which include random fields characteristics, comprehension difficulty, programming, variable slopes and roadways incorporation, and long CPU times [Achireko and Frimpong, 1996; Dowd and

Onur, 1993] in a 3-D geological setting still persist and require continuous research for better solutions.

1.2 Statement of Problem

In general practice, mineralized shallow or outcropping deposits that are economic are exploited using surface mining technology. The development of an operating strategy to exploit such deposits involves designing, optimizing and scheduling the ultimate pit. Open pit mining can be accomplished by an infinite number of paths/steps, each generating a unique net present value. Some mine operators thus optimize pits under some defined assumptions. Whenever any of these assumptions change, the pit optimization process is repeated. The mining sequence normally selected is the one that maximizes the net present value of the entire operation subject to operational constraints. The ultimate pit is the size and shape of the resulting pit at the end of the mine life. It is essentially the optimal ultimate design or pit configuration that represents end-of-life mining limits under assumptions of instantaneous mining conditions. Indeed an optimized open pit is the final dimensions of that open hole (reclaimed or left open) in the ground at the end of the mining period, which yields the maximum recovered metal or profit or net present value for the mine operator.

The dominant methodology for 3-D open pit mine optimization is the graph theory algorithm employed by Lerchs and Grossmann (L-G) in 1965. This algorithm utilizes a 3-D block model under known mineral values, constant costs and constant pit slopes constraints. Though the Lerchs-Grossmann's algorithm has become the standard optimization methodology in open pit design, it does not take into consideration the random field properties (encountered in mine design/optimization). It is difficult to incorporate constraints like variable slopes and haulage roads into the pit optimization. The difficulty encountered in programming the algorithm together with the long CPU times has contributed to a reduction in its effectiveness [Dowd and Onur, 1993]. Over the years, many algorithms namely moving cone (heuristic), linear and dynamic programming, graph theory, stimulation, parameterization together with their modifications have been developed to either resolve the open pit optimization problem or provide modifications/improvements to the other algorithms especially that of Lerchs and Grossmann [Braticevic, 1984; Dowd and Onur, 1993; Gauthier and Gray, 1971; Kim, 1978; Koenigsberg, 1982; Lemieux, 1979; Robinson and Prenn, 1973; Shenggui and Starfield, 1985]. Frimpong and Achireko (1996) departed from the heuristic and classical methods when they developed a back-propagation neural network algorithm to optimize a 2-D open pit. Aside from the fact that the open pit optimization problem is a 3-D problem, the use of a back-propagation could result in a local optimum instead of the global optimum. Their approach took into consideration the stochastic nature of the open pit optimization process. The discovery of an efficient algorithm, which truly optimizes a 3-D open pit design, still remains a fundamental challenge. A 3-D pit is defined in the x-, y- and z- coordinate system, whereas a 2-D pit is defined only in two of the coordinate systems (x-y or x-z or y-z).

It must be emphasized that the generation of an optimized pit layout of a typical open pit mine in 3-D space involves parameters (such as ore reserves and grades, costs, commodity prices and pit wall slopes) which are controlled by stochastic processes. The 2-D neural network algorithm, unlike most of the other algorithms, did not neglect the stochastic nature of the state variables. Most of the pit optimization algorithms that have been developed to date have yielded sub-optimal pits in some cases, resulting in considerable error and uncertainty in defining pit layouts, equipment requirements and mine plans (long-, medium-, and short-term). This might lead to poor investment and inaccurate mine design, development and production decisions.

Over the years, deterministic rock mechanics techniques have become the dominant methodologies employed in characterizing the ore body and the host rock and in designing the ensuing pit wall slopes. Due to the variation in geology and other factors, the design of pit walls could be different from region to region and even in the same region, may be different from location to location. Therefore the use of constant slopes in a mine design may result in the assumption of considerable risk. As a common practice, researchers have employed various probabilistic approaches in slope stability analysis and design [Piteau and Martin, 1977; Kim et al., 1978, Major et al., 1978; Wyllie et al., 1979; Baecher et al., 1980; Priests and Brown, 1983; Roberds, 1990 and 1991; Genske and Walz, 1991; Wylli, 1992; Sandroni, 1993; Duzgun et al., 1995; and Duzgun et al., 1998]. These developments indicate that variable pit slopes may have to be used over the entire mine, which make the results of algorithms under constant pit slopes unreliable.

The Lerchs-Grossmann's 3D optimization algorithm presents a tight choice between a true optimizing algorithm characterized by large computing requirements and a non-optimizing algorithm with lower computing demands and thus fails to yield a truly optimized pit [Kennedy, 1990]. The solution of the Lerchs-Grossmann's 3D optimization algorithm involves the inversion of matrices with large dimensions and long iterative processes, which result in long computing times. Any future changes in the variables require complete re-run of the optimization algorithm and the preparation of new production plans [Frimpong and Achireko, 1996].

It is not necessary for engineers designing a pit to have a detailed knowledge of the mathematics involved in the Lerchs-Grossmann's algorithm that is embodied in a verified software. However its complexity is often advanced as a reason for not using the algorithm [Dowd and Onur, 1993]. The comprehension of the underlying theory for the implementation of the algorithm will help engineers and researchers to factor in and quantify the effects of various combinations of identifiable variables. These variables are regularly encountered by mine engineers, mineral venture planners, evaluators and investors in the evaluation and assessment of economic potential of a mineral deposit.

1.3 Objectives of the Study

The study is aimed at designing a 3-D open pit mine using machine learning and then employing simulated annealing and machine learning (neurogenetic) algorithms to optimize the pit. The design phase of an open pit mine is equally important and is therefore accorded considerable attention in this work. As part of the design process, a machine learning algorithm will be applied to model block grade values. The entire design and optimization process can be broken down into the development of (a) a three-dimensional block model of a mineralized deposit (gold) using geostatistics, machine learning and decision theory, (b) a slope stability model by machine learning or statistics, (c) an economic model of the mine, and (d) a mathematical open pit optimization model by using simulated annealing and machine learning. The results of this new optimization algorithm will be compared to Lerchs and Grossmann's 3-D optimization algorithm.

1.4 Scope and Limitations of the Study

This study deals with the development of design and optimization algorithms for an open pit mine using machine learning. In the open pit design phase, a 3-dimensional block model was developed using geostatistics. Machine learning algorithms, namely adaptive logic networks and a neurogenetic optimizer, are then employed to develop a block predictor. The block models are for both gold and coal deposits. Gold and coal deposits are used as examples because of their different depositional environments and processes. The machine learning algorithms are also employed in developing an optimized slope design. The open pit optimization process is concerned with the definition of a layout which maximizes the net value, at any given time, of an open pit section under technical, operational, economic and safety constraints. Due to the stochastic nature of the block values, the optimization algorithm must take this into account during the optimization process. The approach to be used is a combination of simulated annealing and neurogenetic algorithms.

Mathematical and computer models are developed for the open pit design and optimization process. Where there is an appropriate software, it is used in place of a new code. In the event that appropriate software cannot be located, a code is developed in C/C++, Matlab, GSLIB, Whittle 3D, AutoLISP and visual basic to solve the models. Experiments are run using computer models of the algorithms and validated using data from the Star Gold open pit mine and a coal mine. An intelligent block model and optimized pit model will be developed as the final products of the study. The Lerchs-Grossmann's 3-D algorithm will also be used to optimize the same open pit mine. The results will be analyzed and numerically compared to the intelligent algorithm. Even though some of the concepts may be applicable to other surface mines, this study is limited to open pit mining and is tested using data from the Sabi Gold Project only.

1.5 Research Methodology

The initial part of this thesis reviews the work that has been done on the design and optimization of open pit mines. This is then followed by the acquisition of input information/data and domain knowledge and the use of mathematical models to handle the data. Most of the domain knowledge and data so collected fall into four broad categories, namely geological/grade, geomechanical, economic (price and cost) and other data/information. The starting point of the design process is the use of statistical, geostatistical techniques and decision theory to model and optimize the 3-D block dimensions, assign grades and tonnages to the blocks and calculate the precision and accuracy of the modelling process. Mathematical equations are used as representative models of the geological processes underlying the ore mineralization process. The simulated solution of the geostatistical model results in an optimized 3-D block model representing the values of ore, waste and air (economically equivalent values of ore and waste).

The geomechanical characteristics and data are employed in the development of an appropriate failure model. Using statistical methods, simulation and appropriate factor of safety criteria, the slopes of the various sections of the pit wall are defined. Prices and costs are also estimated. Mathematical models representing the stresses, strengths and other characteristics of the rock mass are employed in determining the allowable slope angles and factor of safety of the various sections of the pit. Due to the stochastic nature of the processes underlying rock characteristics, machine learning, probabilistic risk or reliability analysis will be used to quantify the variability. The reliability index can be used to capture both the mean safety margin and its variability.

In the final phase of the thesis, simulated annealing and neurogenetic algorithms are employed in solving the open pit optimization problem. The simulated annealing and genetic algorithms employ random search methods in optimization and can therefore cater for the random properties of the parameters. The results of the various mathematical (design) models are fed into the "summer" (Optimizer). The optimizer is essentially an intelligent mathematical algorithm employed in learning and predicting the combination of blocks which will yield an optimized ultimate pit configuration under the slope constraints. The actual optimization process involves the use of machine learning algorithms to determine a set of ore and waste blocks that maximize the net value of the entire set of blocks (operation) when mined together over the life of the mine. These set of optimized blocks are subject to pit slopes, operational costs, price of the final product and mine feasibility constraints. The output of the Optimizer will then be subjected to sensitivity analysis until an optimized pit is obtained. The machine learning algorithm to be developed will be compared to the Lerchs-Grossmann's algorithm.

In this work, the author will develop machine learning algorithms to address the stochastic properties of the optimization parameters. Geostatistical techniques and decision theory will be used to develop a block model, assign grades and economic values to the resulting blocks of a typical mineralized gold deposit. Price, cost and slope models will also be built. All these models will be fed into an optimizer called the "summer", which is a machine learning optimization algorithm. The results will then be subjected to sensitivity analysis to yield an optimized 3-D pit.

According to Simon [1981], learning is an adaptation process aimed at improving a system's efficiency and effectiveness on some class of problems. An improvement in efficiency means that the open pit optimization problem is solved with fewer computational resources such as processing time and memory space. Supposing there is an acceptable solution space for the open pit optimization problem, improvement in the solution quality entails narrowing the solution space that the system or algorithm produces so that they satisfy some set of solution-quality criteria. Narrowing the solution space could also lead to a reduction in the error variance and the accompanying operating risk. A third dimension of improvement is the incremental extension of the population of problems that the system can solve. Thus for an intelligent system, the three dimensions of improvement of an algorithm is :

1. an improvement in the efficiency of the process of problem solving,
2. an improvement in the quality of the solution so produced,
3. an increase in the population of problems that the algorithm may solve.

Note must be taken of the fact that a learning algorithm for one type of performance improvement may not necessarily be extended to other classes of problems.

1.6 Contributions and Significance of the Study

The open pit mine optimization problem has plagued the mineral industry for decades. A truly optimized open pit affords the mine operator the ability to determine equipment sizes, number and type of equipment, mine personnel levels and a host of other issues involved in the smooth running of an open pit mine. This study will replace the existing 2-D intelligent back-propagation algorithm by Frimpong and Achireko (1997) with a 3-D adaptive logic network and/or simulated annealing cum neuro-genetic algorithm. The study charts a new course for developing the theory of open pit design and optimization that incorporates the random field stochastic processes. The back-propagation (BP) algorithm may converge to a local optimum instead of a global optimum. BP may also generalize the learning process over the input space and may therefore output wrong results in some areas.

The adaptive logic network (ALN) algorithm employs a decision tree to partition the input space so that a small amount of linear pieces will have to be evaluated to obtain an output. Thus ALNs allow the input space to be partitioned, a feature which may be useful in a pit with different mineralogical and/or geomechanical settings. ALNs use hard limiters instead of sigmoids together with logic AND or OR above the first hidden layer leading to fast training and evaluation. Though it is difficult to check what the BP algorithm is doing (black box phenomena), an ALN system could be checked for safety. The design procedures and the algorithms employed in this work will take into consideration the stochastic nature of mineral grades, price, costs and pit wall slopes. The architecture of the neurogenetic algorithms will enable the development of intelligent block models, which can be used to predict block values at any point in the pit. Simulated annealing and neurogenetic algorithms will be very useful in the open pit optimization process. These algorithms do take the stochastic nature of the parameters into consideration. The resulting optimized pit will therefore be the best pit under the underlying constraints. This will reduce the risks associated with the mine investment and provide engineers reliable information for mine planning and equipment scheduling. All these will lead to reduced cost per unit product and higher financial gains for the shareholders.

As in the case of the Lerchs-Grossmann's algorithm, the stochastic nature of the ore grades and the resulting reserve calculations are accounted for in the geostatistical modelling process. A measure of model correctness will be employed to ensure the ore reserve model is appropriate within the given modelling constraints. The level of statistical, geostatistical and systematic uncertainty (e.g. head grade, tonnage, etc) as well as the risk (probability of loss or gain) associated with the numbers are modelled in order to calculate some degree of confidence, the global reserves and mine plans. The resulting block values will lead to the minimum dilution and the lowest level of throughput variability of the run-of-mine ore at the processing plant intake. The machine learning algorithm will also take into consideration the stochastic nature of such parameters as price, cost, slopes and grade of the ore block. The algorithm will solve the back and forward passes required in the Lerchs-Grossmann's algorithm.

In this era of dwindling profits, environmental awareness, tight financial resources, the industry is looking for answers to the optimized pit problem. This work employs a machine learning algorithm to resolve the open pit optimization problem. It takes into consideration the stochastic nature of the input parameters, which has not been fully addressed to date.

1.7 Report Structure

Chapter 1 of this thesis discusses the background, problem statement, objective of the study as well as its scope and limitations. Other sections of this chapter, are contributions and industrial significance of the study, the research methodology employed and the report structure. The first part of Chapter 2 provides an overview of open pit mine design techniques which comprises the various failure modes, geometrical and computer-aided designs employed in an open pit mine design. The second section of this chapter deals with the mine optimization methods that have been employed in the mineral industry over the past forty years.

At the beginning of Chapter 3, the open pit design-modeling paradigm employed in this thesis is fully outlined. In this section, which is essentially on the theoretical foundations of the design/modeling process, issues such as hypothesis testing, model correctness, accuracy and optimality criteria will be discussed. Other issues in this chapter include the open pit mine design block model, the economic value of a block, and the constraints and stochastic processes involved in the optimization parameters. The input parameters of the open pit optimization process are mathematically modelled and evaluated to yield values to be used in the optimizer. These parameters are rock slope, ore reserves/block modelling, cost figures and prices. Chapter 4 is mainly concerned with the machine learning algorithms employed in the optimizer. Mathematical representations of the dominant algorithms will be given in the final work.

Chapter 5 deals with computer modelling/programming of the adaptive logic network and simulated annealing and machine learning (neurogenetic) algorithms. The machine learning algorithms under consideration are continuous adaptive time neural networks, generalized regression neural networks and multi-layer feedforward (back-propagation) neural networks. These neural networks will be modelled using genetic algorithms. This process involves the development of modelling concepts and philosophies, data structures, code and a process of verification. Most of the programming will be done in visual C/C++ in the case of adaptive logic network (ALNBench) and C in the case of simulated annealing. Other appropriate software like the neuro-genetic optimizer (BioComp) and Geostatistical Software Library (GSLIB) are also used. In Chapter 6, various experiments are employed in determining the validity of the algorithm and the results are compared with the L-G algorithm. The experimentation, testing, and validation of the models will be done using a Star Gold Mine case study. The results so obtained will be compared to the Lerchs-Grossmann's to investigate whether there are any differences and the reasons for these differences. Intelligent neurogenetic block models for open pit gold and coal mines will also be developed and used to predict block values. In Chapter 7 the results of the experiments are analysed and discussed. Depending on the degree of confidence that can be assigned to the results obtained, one of the machine learning methods may become the method of choice in open pit optimization. The thesis will then be concluded in chapter 8 with a discussion of the major findings of this work as well as some recommendations for further work.

CHAPTER 2

DETAILED LITERATURE SURVEY

An extensive literature survey is the focus of this chapter. This literature survey is being employed in evaluating past and current developments in the design and optimization of open pit mines. Open pit mine design can be divided into geomechanical mine design, geometrical mine design and computerized mine design. Orebody modelling and evaluation (block modelling) is considered as an input into mine design.

2.1 Geomechanical Mine Design

The stability of an open pit mine slope is of major significance to the safety of personnel and equipment working in the confines of the pit. The dichotomy however is that cutting the flatter will increase safety and the amount of waste excavated to mine a given amount of ore. In practice the ultimate slope of the pit is cut to the steepest possible angle so as to minimize the amount of waste material excavated.

Depending on the geotechnical characteristics of the pit walls and the perceived instability, the analysis of the slope of a potential open pit mine is two-fold namely stability evaluation on the bench scale and the full-height scale. It is impossible to predict the exact failure geometry and modes due to the uncertainties and natural variability in rock masses with many orientations of discontinuities. However, simplified instability model (dictated by the type and structural control), have been employed over the years to predict the slope geometry of open pit mines. Some of the common forms of instability models, which are amenable to engineering analysis are wedge failure, planar failure, toppling failure, circular failure and special failures [Barron, 1997]. The most important thing is the degree of certainty of the known geological information.

The stability analysis of these modes of failure are either discrete or probabilistic. The discrete method of analysis is dependent on an acceptable factor of safety. The factor of safety is defined as the ratio of the resisting forces to the shearing/disturbing forces. A factor of safety between 1.25 and 2.00 is acceptable for geotechnical designs in open pit mines [Barron, 1997]. Unfortunately the computation of a reliable factor of safety requires the specified rock properties and accurately computed resisting and shearing/disturbing forces. These conditions are not easily achievable. Each of the variables employed in a typical geotechnical design has its degree of uncertainty associated with it. A design method that has been used in recent times to address this problem is the probabilistic approach. This approach employs the coefficient of reliability or the probability of failure as the decision criteria. A characteristic probability distribution function is assigned to each variable. The factor of safety is calculated from a random value which is selected from each of the probability distribution functions [Turner and Schuster, 1996]. The random numbers assigned to each of the variables are generated from a Monte Carlo simulation. The process is repeated a large number of times (100 or more) to develop a cumulative density function of the factor of safety. The coefficient of reliability (%) or the probability of failure (100 - coefficient of reliability) can be calculated together with the economic consequences of failure. This approach can lead to a better decision on slope geometry than the single factor of safety (discrete) approach.

Matthews and McTaggert (1969), in their analysis of the 1965 Hope landslide of British Columbia, introduced stability analysis of the planar failure mode. Hodge and Freeze(1977) documented the influence of ground water on planar failure. Sarma (1979) employed limit equilibrium analysis which involved the method of slices to analyse planar failure. Hoek and Bray (1981) developed graphical methods applicable to planar failure. Plane shear failure is the most common and costly failure mode since it could cover a larger area and involve high volume of mass. It occurs along discontinuities or cracks which run parallel or sub-parallel to the slope face.

The assumptions underlying the plane shear failure with a tension crack are that the tension crack and shear plane strike parallel to the slope face. Water enters the sliding surface at the base of the tension crack, seeps along the failure plane, leaving the plane at atmospheric pressure. It is also assumed that release surfaces are present and there are no rotational forces. The weight, uplift and disturbing forces from the water all act through the center of the block and there is no lateral resistance to sliding at the ends of this system. The shear strength of the sliding surface is governed by Coulomb - Navier failure criterion. A unit thickness is used as the basis for the calculation of the factor of safety of the plane failure mode. The weight of the block can be calculated for the tension crack beyond the crest as well as in the slope face.

Piteau (1972) indicated that rock mass lithology and structure are the main elements controlling wedge failure. Wedge failures are the result of rock masses sliding along two intersecting planes or discontinuities which are at oblique angles to a slope surface. The litho-structural properties of the rock mass dictate the formation and occurrence of wedge failures. The structural conditions governing the occurrence of wedge failures are that;

(i) the dip direction of the slope face is approximately equivalent to the trend of the line of intersection of the two planes,

(ii) the line of intersection daylights in the slope face (that is the dip of the slope face is greater than the plunge of the line of intersection) and

(iii) the average angle of friction of the two planes is less than the plunge of the line of intersection.

Stereographic analysis or Marklands test is employed to determine whether sliding will occur on both planes or on a single plane [Hoek and Bray, 1981]. Another test is the Hocking's test for the direction of the sliding of a wedge. A wedge can be formed such that it slides on one plane when it moves, loosing contact with the other plane. According to this test, if the dip of either plane falls between the dip direction of the slope face and the trend of the line of intersection, sliding will occur on the plane rather than along the line of intersection. Kinematic analysis is normally followed by a more rigorous stability (rigid-body) analysis which assumes that failure occurs by linear sliding along either one of the planes or on the line of intersection of the two discontinuities. In the wedge failure mode the factor of safety can be calculated for the following situations: (i) a tension crack is absent but the slope face is saturated with water, (ii) the slope is fully-drained, (iii) the only active forces are due to friction, and (iv) the friction angle is the same for both planes.

Goodman and Bray (1976), Hittinger (1978), Choquet and Tanon (1985) and Willie (1992) have analyzed toppling failures using the limit equilibrium methodology. Rock masses susceptible to toppling failures are normally divided into a series of slabs or columns emanating from a set of fractures striking near-parallel and dipping steeply to the slope face. Rotation of the rock slabs or columns about a fixed point near to or at the slope base vis-à-vis slippage between the layers result in toppling action. Bedded or foliated metamorphic, sedimentary and columnar basalts are very prone to this type of failure. Hoek and Bray (1981) described several types of toppling failure, namely flexural, block or a combination of the two. A fundamental condition for the occurrence of toppling failure is that the centre of gravity must fall outside the base of the slab or column. Failure is only possible when shear failure occurs at the base of the slabs.

According to the methodology of Goodman and Bray (1976), the conditions necessary for toppling failure are that: (i) the dip of the layers must be between 160 and 200 degrees to the dip of the slope face (the strike of the rock layers and the slope face must be parallel or near parallel with a 20 degree difference and the layers must dip into the slope), and (ii) the plunge of the normal to the top of the slope face must be less than the slope face inclination less the friction angle. Choquet and Tanon (1985) proposed various modifications based on extensive nomogram analysis. They defined the factor of safety for toppling failure. Cundall (1987), Lorig(1991), Byme, and Hammet have employed computer-aided numerical techniques to model toppling processes but they are yet to be well developed for general engineering use. Using limit equilibrium analysis, the force required to prevent toppling of the nth block can be calculated for a set of blocks in a formation.

Circular failure or circular arc (or rotational shear) failure occurs in very weak rocks, soils, weathered material and in heavily-jointed rock masses (e.g. a waste dump). The failure surface is a circular or curvi-linear surface of minimum shear resistance with respect to the disturbing forces. The failure surface may or may not pass through the toe of the slope. The rock masses subjected to circular failure are so highly fragmented that the failure surface would not have to go through the hard part of the rock mass. Circular arc failure in rock is essentially similar to classical rotational failure in soils, the only exception being that the failure surface or circle in rock tends to be shallower and has a larger radius.

Kinematic analysis, as performed in circular failure, is aimed at eliminating other modes of failure. Thus, if no persistent structures are present, circular failure can be employed in slope design. tability analysis in circular failures is dictated by the shear-strength criterion employed in the characterization of the rock mass. Techniques developed for soils are employed if the rock mass satisfies the Mohr-Couloumb criterion. Such an analysis involves stability charts [Duncan 1996 and Hoek and Bray 1981] and analytical techniques of Janbu (1954), Bishop (1955), Morgenstern and Price (1965) and Sarma (1979). If the rock mass is very dilatant as a result of strong intact strength of the individual blocks, a non-linear shear strength criterion is employed in the analysis. Though the failure modes described above are the most common, other modes of failure like the column buckling failure and bilinear wedge failure have been known to occur. Generally, column buckling failure occurs in slope faces that are masked by parallel, steeply dipping discontinuities. Column buckling failure is very similar to planar failure except that the discontinuity forming the failure surface, does not necessarily daylight. Goodman (1980) and Cavers (1981) analyzed this mode of failure using the classical method of column stability where Euler's critical stress for buckling is equivalent to the stress parallel to the axis of a column. In the bilinear wedge failure, the upper block is separated from the lower block by a steeply dipping discontinuity, which enable the block to move relative to each other. Such failure surfaces are major discontinuities for instance a fault. Generally, the dip of the lower block is lower than the friction angle of the surface whereas the dip of the upper block is higher than the friction angle. The kinematic analyses of this mode of failure are similar to planar failure.

In slope stability design and analysis, a factor of safety can be regarded as an accurate representation of our belief in the design if the material characteristics, stress and strength of the rock mass can be computed at an acceptable degree of accuracy. Unfortunately, the variables involved in a design have their own degree of uncertainty. Thus, the factor of safety can deviate from our expectations, jeopardizing mine equipment and personnel. The reliability coefficient / failure probability can be used as an index for the factor of safety.

2.2 Geometrical Mine Design

The geometry of every open pit layout may be different from the others because mineral deposits have different sizes, shapes, orientations, depths of occurrence, mineralogical composition and geotechnical characteristics. Obviously, material characteristics and equipment technology will influence the pit geometry. The sequencing of the geometry is very important if the desired economic results are to be achieved. A flatter slope leads requires the excavation of additional waste material which will lead to higher mining cost. Slopes in open pit mines are designed to perform specific functions.

The three major components of an open pit slope design are: bench configuration, inter - ramp angle and overall slope angle. According to Call and Savely (1990) and Keaton and Beckwith (1996), designing open pit slopes involves the following; (i) specification of the functional requirements, which normally is provision of safe access with acceptable maintenance and economy, (ii) determination of design sectors and preparation of appropriate layouts to satisfy the functional requirements, (iii) enumeration and preliminary design of all the solutions which could possibly satisfy the functional requirements, (iv) stability analysis of the chosen design(s) to estimate probability of failure and expected volume of failed material for bench, inter - ramp and overall slopes, (v) development of maximum inter - ramp slopes based on catch bench criteria, (vi) using a cost-benefit, functionality and safety criteria to determine optimum pit slopes and (vii) detailed designing of the optimum pit slopes.

A number of slopes are required as part of the design of the optimized open pit. The optimized pit slope is a layout that ensures the maximum economic recovery from an open pit. As a general practice, the final slopes of an open pit mine is excavated to the steepest possible angle in order to reduce the amount of waste material (soil and rock) excavation to the minimum, during the recovery of the ore body. The design engineer must however bear in mind that the cost of a single major slope failure in terms of loss of production, equipment and human life, could totally offset any financial gains. The design engineer must therefore deal (quantitatively and/or qualitatively) with the consequences of failure in the pit design process. The best economic design may be to allow the slope to fail and clear up the debris if the production and equipment are not at risk. This does not occur in reality due to the cost and consequences of failure. A failed slope may disrupt production and trigger safety investigations. Slopes are therefore not allowed to fail during the operating life of the mine. The geometry of the pit depends on the bench configuration (bench height, bench width, type of benches), loading system, production strategy, deposit characteristics, equipment used and regulatory requirements.

2.2.1 Bench Design

The bench height is considered as the vertical distance between each of the horizontal levels of the open pit. Generally, all the benches of an open pit mine have the same height, unless geological and different equipment conditions dictate otherwise. The bench height is affected by ore grade or dilution, the required degree of selectivity of ore from waste, production rate, physical characteristics of the deposit, equipment type and size, climatic conditions and safety of personnel and equipment.

Though the inclination is to design the bench height as high as possible, the size and type of the loading equipment for the production desired may limit the bench height. For a truck - shovel system, for instance, the teeth of the shovel dipper should be able to scale the crest of the working bench, in order to prevent hang-ups in the upper section of the pit wall and to achieve a smooth wall surface. Assuming that the mining equipment is the defining factor, the height of the bench is equivalent to the boom point of the shovel and is given by the equation;

$$\text{Bench Height, } B_h = \text{Boom Point of Shovel} \qquad 2.1$$

2.2.2 Bench Width Design.

In a typical open pit mine, there are several types of benches - working bench, safety bench and catch bench. Depending on the functions of the bench, there are varying bench widths. A typical working bench consists of a safety berm, minimum working room for the shovels and trucks and any other safety or regulatory requirements. A safety berm is required to augment the safety of the operations. According to Hustrulid (1990), the width of the working bench is given by;

$$b_w = b_{rw} + c_r + n\frac{TW}{2} + RCR + DOR \qquad 2.2$$

2.2.3 Width of Safety Berm

The height of a safety berm should be at least equivalent to the height of the haulage or largest truck. The width of the safety berm b_{rw}, is given by

$$b_{rw} = \frac{2h_a}{\tan r} \qquad 2.3$$

In the case of single back-up loading, the working bench width is defined as;

$$b_w = b_{rw} + 1.6 + \frac{TW}{2} + DOR + RCR \qquad 2.4$$

For a double back-up loading system, the working bench width is given by;

$$b_w = b_{rw} + 2*1.6 + 2*\left(\frac{TW}{2}\right) + 2*DOR \qquad 2.5$$

A safety or catch bench serves as a ditch for catching falling rocks, small volumes of failed slope material, especially those resulting from wedge failures. According to a report published by CANMET(1977), the volume of failed material for an open pit bench can be considered as half the cell volume. Assuming wedge failure and no bulking, the width of the catch bench is therefore given by;

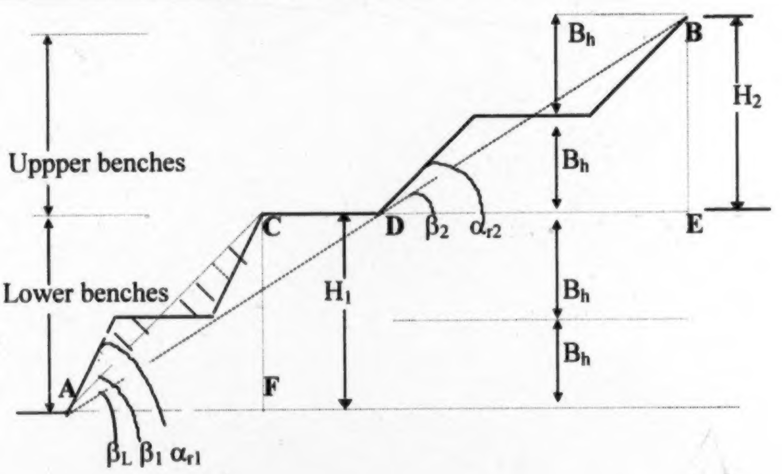
$$b_c = b_{rw} + b_i = \left(\frac{2h_a}{\tan r}\right) + \frac{B_h}{\tan \varphi_p} * \sqrt{\left\{\frac{(\tan \alpha_{ri} - \tan \varphi_p)(\tan \varphi_p - \tan r)}{\tan r \tan \alpha_{ri}}\right\}} \qquad 2.6$$

The volume of failed material is given by,

$$V_m = \frac{B_h^3}{4}\left\{\frac{1}{\tan\varphi_p} - \frac{1}{\tan\alpha_n}\right\} \tag{2.7}$$

The mass of material involved is given by;

$$W_m = \rho * V_m \tag{2.8}$$



B_h
B
H_2
Uppper benches
B_h
C
D
β_2
α_{r2}
E
Lower benches
H_1
B_h
B_h
A
F
β_L β_1 α_{r1}

Figure 2.1 Pit Slope Design Without A Ramp

2.2.4 Imposition of Slope Walls

The major component of the slope design is the maximum angles of the slopes. These angles which depend on the geotechnical characteristics of the various areas of the mine, could be different for the different geological areas. It is assumed that limit equilibrium analysis will be sufficient to predict the stability of the slopes if the number of faults in each geological area is less than or equal to 3. Assuming the entire pit could be divided into two geotechnically different vertical sections as depicted in Figures 2.1 and 2.2. The angle of the ultimate pit slope for the lower section (1) can be calculated from the equation,

$$\tan\beta_1 = \frac{2B_h}{2B_h\cot\alpha_{r1} + b_1} \qquad 2.9$$

Similarly, the angle of the ultimate pit slope for the upper section (2) can be calculated from the equation,

$$\tan\beta_2 = \frac{2B_h}{2B_h\cot\alpha_{r2} + b_2} \qquad 2.10$$

Thus the ultimate pit slope angle for the nth section of multi-slope (variable) pit can be calculated from the equation,

$$\tan\beta_n = \frac{2B_h}{2B_h\cot\alpha_{rn} + b_n} \qquad 2.11$$

Lerchs-Grossmann's ultimate pit angle for the two-slope pit can be calculated from,

$$\tan\beta_L = \frac{2(2 * B_h)}{\{2B_h(\cot\alpha_{r1} + \cot\alpha_{r2}) + CD + b_1 + b_2\}} \qquad 2.12$$

and for an n slope open pit mine,

$$\tan\beta_L = \frac{n(2 * B_h)}{\left\{2B_h\sum_{k=1}^{n}\cot\alpha_{rk} + \sum_{k=1}^{n} b_k + CD\right\}} \qquad 2.13$$

When the sectional slope angles are used for each of the geotechnically-defined sections of the ultimate pit wall, mass balance is achieved and the ultimate pit limit is given by line ACDB (see Figure 2.1). The pit wall employed by Lerchs and Grossmann in their model is defined by line AB (see Figure 2.1.

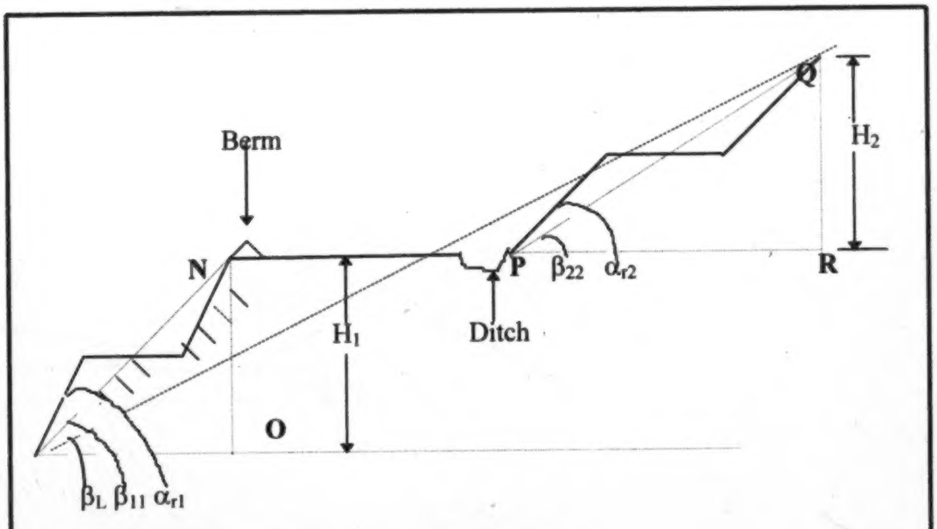


Figure 2.2: Pit Slope Design With A Ramp/Road

Thus the total depth (H_t) of an open pit with n benches of the same bench height is given by;

$$H_t = n * B_h \tag{2.14}$$

From Figure 2.2, considering NP as the ramp or road, the slope angles for sections 1 and 2 can be calculated from;

$$\tan\beta_{11} = \frac{2B_h}{2B_h \cot\alpha_{r1} + b_1} \tag{2.15}$$

$$\tan\beta_{22} = \frac{2B_h}{2B_h \cot\alpha_{r2} + b_2} \qquad 2.16$$

The ultimate pit outline then becomes MNPQ, where NP is the ramp (see Figure 2.2). Lerchs-Grossmann's ultimate pit angle is given by,

$$\tan\beta_L = \frac{2(2 * B_h)}{\left\{2B_h(\cot\alpha_{r1} + \cot\alpha_{r2}) + b_1 + b_2 + NP\right\}} \qquad 2.17$$

and for an n slope open pit mine,

$$\tan\beta_L = \frac{n(2 * B_h)}{\left\{2B_h \sum_{k=1}^{n} \cot\alpha_{rk} + \sum_{k=1}^{n} b_k + t(NP)\right\}} \qquad 2.18$$

The ultimate pit outline then becomes MNPQ, where NP is the ramp (see Figure 2.1). In the L-G approach, the ultimate pit outline is line MQ. Once again the L-G pit outline involves extra material as compared to the sectional approach. However, in both approaches, the upper walls of the pit have been substantially pushed back due to the introduction of the road, NP. This will lead to higher overburden volumes and lower returns to the mine operator as stripping ratios will increase.

Haul roads could be designed in areas where slope angles are gentler and or stability problems are minimized. The size of the pit may warrant a zigzag layout as the spiral layout in larger pits may affect haulage distance to a great extent. The tightness of the pit wall may also only allow for a zigzag road layout. High slope angles of the pit walls, on the other hand may not allow for a zigzag layout as the amount of requisite excavation may unreasonably increase the stripping ratio. Hazards due to slope instability could prevent the placement of haul road on that side of the open pit. This is especially true in the part of a pit where high water pressures exist or the rock mass is characterized as poor. In the latter case a spiral layout is recommended. The length of the haul roads should be minimized as much as possible in order to reduce construction costs and travel time. A haul road must provide the safest, quickest and cheapest access to the mining operations in the pit. However regulatory and production constraints may prevent the placement of haul roads in certain portions of the open pit operation due to road slope limitations and or surface disturbance or regarding requirements.

2.2.5 Imposition of Haul Road

When the haul road is located outside the orebody, the volume of waste and ore to be excavated are increased which may affect the economics of the mining operation. On the other hand, the location of a haul road within the ore body may reduce the waste material as well as the mineable ore, which may not be advisable in high-grade deposits. Atkinson (1983) has indicated that the placement of haul roads are site-specific and are dependent on the economic and geotechnical conditions within the open pit mine in question. In this work such an analysis will be done in the final design to optimize the haul road design at minimum costs.

2.3 Computer-Aided Mine Design

The Babylonians built the first mechanical computer but it was not until the advent of the transistor in 1948 that the technology advanced. In the 1950's and 1960's, auto, aircraft and some few large companies begun building their own computer-aided drafting (CAD) systems. Ivan Sutherland's (1963) Sketchpad doctoral thesis resolved all the graphical human interface issues. In the 1970's as computers became smaller, more powerful and cheaper, turnkey CAD systems combined software and computers to resolve 2-D drafting problems. The debut of the computer 32-bit midi architecture in the 1980's greatly enhanced CAD software development. Mike Riddle wrote MicroCAD in 1980 and John Walker was hired to form Autodesk which marketed the product as AutoCAD in 1984. A lower priced personal computer was also introduced to the market in the 1980's. In 1986 Keith Bentley formed Bentley Systems Inc (later known as Intergraph). In the 1990's less expensive and more powerful desktop computer units abound and a new drafting paradigm, Pro/Engineer was born. Computer-aided drafting matured into computer-aided design. New constructs like solids modelling, complex 3-D geometry, parameterization, animation, virtual reality and constraint-based modelling (Dimensional Constraint Modelling) were added to computer-aided design (CAD) systems. In the new millennium, intelligent systems may employ GIS data and/or limited information to generate accurate solids and 3-D geometry in a virtual architecture.

Computer-aided open pit mine planning dates back to 1960 when a group of mining engineers at U.S Steel Corporations Minnesota operations employed an IBM 650 to (Axelson, 1964). They simulated mining, beneficiation and ore shipment using a series of mathematical models programmed into the computer. The results were then employed in defining the economic mining limits (ultimate pit limit). Computer-aided design evolved with the growth and the ease of computer usage. In the 1960's most of the software that were developed were programmed into mainframe computers. There were mostly command-line driven and modular with proprietary databases. The 1970's witnessed a migration to mini computers and software were written in Fortran. Layered menus with access to independent modules were first introduced. In the 1980's demand for mineral products increased (gold reached a peak price of $800+ per oz.) and small mining software companies were born. Some of the software were ported to personal computers. Graphics and some commercial databases were introduced. Some optimization programs were introduced. In the 1990's graphics have been vastly improved due to the availability of higher computing power at lower prices. Most of the software are now written in C/C++ with open database connectivity (ODBC) in a Windows NT or 95 environment. The menus have been replaced by pull-down style graphical user interfaces (GUI's). Simulation programs were introduced in the early part of the 1990's.

The most common computer-based mine design systems available today are vulcan, gemcom, datamine, medsystem, mincom, lynx/microlynx, and surpac. Generally the important modules in these software are drill-hole and/or exploration databases, statistics and geostatistics, geological modelling, open pit and/or underground mine design and surveying. The CAD aspects of these software enable mining engineers to develop geological and block models of orebodies, design various layouts of mining excavations and output the results in diverse graphical formats. The best software for open pit optimization are Whittle's 4D/4X and Earthworks MAXiPiT. They both employ the

Lerchs-Grossmann's algorithm and appear to give identical results when employed in open pit optimization.

2.3.1 Geostatistical Modelling in Computer-Aided Mine Design

Geostatistics is the study of phenomena that fluctuate in space and/or time (Deutsch and Journel, 1998). In mining engineering, it is the application of the theory of regionalized variables developed by Danie Krige (1951) and mathematically formalized by Georges Matheron (1970) to ore reserve modeling and estimation.

According to Journel and Huijbregts (1978), the variogram model, albeit any artificial errors introduced, is a summary of the structural information, which is then employed in reserve estimation. The variogram model is then employed in estimation. A variogram model for some distance and/or direction is required for further analysis of the phenomena under investigation (Isaaks and Srivastava,1989). The variogram quantifies the spatial variability of the contained gold by computing the variance of the grade values measured some distance h, apart. The variogram/variance increases with the separation distance. According to Deutsch (1998), the variogram model so built must be positive definite. In the case of co-regionalization, the model must be positive semi-definite. Generally, models are built as a linear combination of two or more types of basis models. The basic models of regionalization are pure nugget effect, spherical, exponential, gaussian and power models. Besides the power model, all the other variogram models are termed bounded, that is they reach a constant sill at some distance, termed the range.

Kriging techniques are estimation methods that minimize the error variance of estimation (as calculated with the variogram model). Kriging is an estimation technique used for spatially distributed data. According to Cressie (1990), kriging techniques were employed as far back 40 years ago and was applied to mining and other fields [Krige, 1951; Journel and Huijbregts, 1978]. According to Goovaerts (1998) and Journel (1988), kriging is a generalized linear regression used to estimate a continuous attribute (kriging) or two continuous attributes(cokriging). Kriging is considered to be the best linear unbiased estimator (BLUE) of the unknown parameter being quantified (Journel and Huijbregts, 1978). The kriging estimation variance can be affected by errors in the calculation of the variogram.

Theoretically, a sample point or grade is considered as a single realization of the regionalized random function. The class of random functions forming the basis of the phenomena under investigation has a characteristic distribution function and a variogram (covariance) model. The geostatistical model developed to date is one realization (albeit a smoothed version) of this random function in bounded space. Simulation can be used to draw realizations of the random function from the pool, which honor the model parameters including the variogram developed to date. The reproduction of the presence or absence of spatial dependence of the one or several variables is paramount to the success of the modeling and evaluation process.

2.4 Open Pit Optimization Algorithms

The determination of the optimum contours of an open pit mine is a fundamental requirement for an effective open pit mine planning and design. The optimum pit limits so determined, provide a basis for economic evaluation, short and long term mine planning decisions. The ultimate pit limit defines the size and shape of an open pit at the end of its life based on the operating, technical, economic and ground stability constraints. They also determine the extent of mineable reserves and waste materials to be moved in the mining process. Pit limits on the surface mark the boundaries for locating surface structures, such as processing plants and mine offices. The pit limits would normally delineate the limiting boundary beyond which the open pit mining of a given deposit will be uneconomic. As such the pit limits are commonly referred to as the Economic Pit Limits (EPL), Ultimate Pit Limits (UPL) or Ultimate Pit Design (UPD).

Over the years various researchers/workers/mining companies [Lerchs and Grossmann, 1965; Meyer, 1969; Johnson, 1968; Lemieux, 1968; Marino and Slama, 1972; Phillips, 1972; Koborov 1974; Koenigsberg, 1982; Wilke and Wright 1984; Denby and Schofield, 1994; Achireko and Frimpong, 1996] have introduced a host of open pit optimization methods. In the past forty years, optimization techniques that have been used in open pit mining can be classified into industrial engineering or mathematical programming methods (graph theory, dynamic, integer, linear, geometric and network programming, and parameterization), heuristics (moving cone) and artificial intelligence (genetic and back-propagation neural network optimization algorithms).

A number of the mathematical techniques that have so far been employed to solve this problem have themselves posed considerable computational difficulties. These methods have also not adequately addressed the following problems: (i) stochastic nature of ore reserves/grades and mineral prices, (ii) the amount of ore and waste to be mined each year, (iii) the specific blocks to be mined each year, (iv) the mine life which defines the expected time horizon of the operation and (v) the ultimate pit limits which delineate the total reserve to be mined and the final pit shape at the end of the operation. The more elaborate methods use manual, stimulation, linear programming, dynamic programming, graph theory, heuristic and parameterization approaches to the open pit limit design. These methods are limited in their respective approaches to defining the optimum pit limit. Among them, the four rigorous optimizing techniques, that have mathematical proofs, are graph theory [Lerchs and Grossmann, 1965], dynamic programming [Lerchs and Grossmann, 1965], linear programming [Meyer, 1969] and network flow [Johnson, 1968]. Heuristic algorithms [Lemieux, 1968; Marino and Slama, 1972; Phillips, 1972; Korobov, 1974] lack rigorous mathematical proof. In the following sections, various algorithms for open pit design and optimization and their limitations have been described.

2.4.1 The Manual Approach to Open Pit Design and Optimization

Manual approach to open pit design and optimization is basically a trial and error method in which the analyst uses subjective and objective analyses to define the pit limits. This approach requires (i) vertical sections showing clearly the ore boundaries, the grade distribution within the ore, the overburden and the waste rock portions; (ii) plans for each proposed mine level showing corresponding ore and waste details as in (i); (iii) allowable maximum slope angles for the various rock types; (iv) minimum width at the proposed pit bottom; and (v) relevant stripping ratio curves showing the variation of the stripping ratio with the ore grades and possible selling prices. The overall, incremental, periodic and break-even stripping ratios are taken into consideration in the design of the pit. These ratios depict the relative proportion of waste material to ore to be mined and provide the necessary profit information. Stripping curves, the behaviour of stripping ratio with grade variation, are normally constructed to cover possible grade ranges at the pit limits as a pre-requisite for a manual pit design exercise [Pana and Davey, 1973].

With the aid of the stripping ratio curves and the cross-sections showing the ore boundaries and grades, the pit limits will be located on each section as follows: (i) Note the grades around the two ends of the ore on a given level and work out a weighted average grade to include the ore up to the top of the grid blocks; (ii) Construct the stripping ratio curve and note the allowable break-even stripping ratio; (iii) Construct trial slope lines such that the ratio of the intercept along waste to that along ore correspond to the break-even stripping ratio; (iv) The procedures in (iii) are repeated for other sections of the deposit; (v) When the pit limit has been located on each vertical section, the intersections are then transferred to the level plans. The total ore reserve and waste volumes and tonnages are determined using the planimetry method.

This trial-and-error method helps in the development of the basic skills required for the geometrical manipulations. These skills are required in such areas as haul road design and phase plan development, which is needed with fully computerized pit design approaches. This approach has a number of limitations. The manual approach is very time consuming. The assumption that the total recovery during mineral processing is constant in the derivation of the stripping curves would not necessarily hold true for different ore grades. In fact the effective recovery will invariably depend on the grade of the mill feed. This method can only be used for small and/or geologically simple mineral deposit. The method can be tedious and computationally intractable in a massive orebody.

2.4.2 The Moving Cone Algorithm

The moving cone algorithm is one of the algorithms for optimizing open pit limits [Pana, 1965; Williams, 1970; Lemieux, 1968]. The 2-D moving cone algorithm uses the following optimization criteria, deduced from the definition of the open pit design problem, namely: (i) maximization of total pit economic value, (ii) maximization of value per tonne of saleable product, (iii) maximization of the mine's life, provided the value per tonne does not fall below a certain figure, and (iv) maximization of the metal content within the pit [Wright, 1990]. The maximization of the total economic value is by far the most common optimization criterion in open pit mine design. As in other algorithms, this algorithm ensures that before mining any block, all the blocks above must be removed. The minimum removal cone on a block is the cone bounded by the maximum allowable slope angles in all directions, from the block up to the ground surface. The cone on any block is dependent upon the different slope angles for the different materials overlaying the block in the different directions up to the ground surface.

The basic element for the optimization process is the minimum removal cone. A summary of the algorithm can be stated as follows: (i) Start from the surface and search for ore blocks with positive economic block value; (ii) Construct the minimum removal cones on such ore blocks; (iii) If the sum of the economic block values (EBV) of all blocks contained in a given cone, including the ore block in question, is positive, consider the cone removed; (iv) Continue the search until all the positive ore blocks in the block model have been examined; (v) The ultimate pit is formed by the shape left after the removal of all positive valued cones. Suppose a section of an open pit mine is represented by the set of blocks in Figure 2.3.

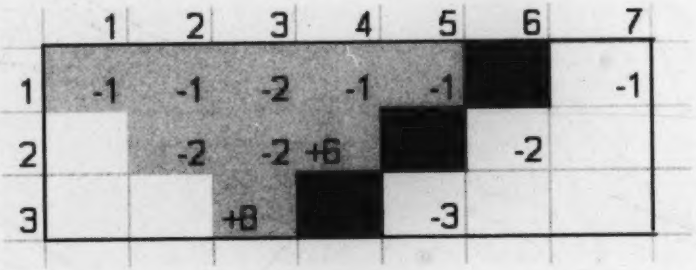


Figure 2.3 Block Diagram for the Floating Method

The following steps are employed in optimizing the pit with the floating cone algorithm.

Step 1. A stepped cone is floated from left to right along the top row. Any positive block (block $_{6,1}$ = +1) is removed.

Step2. The apex of the cone is now moved to row 2 and then 3. The cone is floated from left to right in each case. The values of the blocks removed are summed to determine the profitability of the section. The pit value is given as 5 (-1-1-2-1-1++-2-2+6+8).

The floating cone method can be an expensive process, especially for a large open pit mine. Furthermore, the technique can miss the optimum pit limit under certain unusual conditions, because it cannot recognize a joint contribution by two ore blocks that are laterally some distance apart. Lemieux (1968) stated that his heuristic algorithm overcame this shortcoming of the moving cone technique. However, the geometrical requirements of slope stability in combination with certain ore blocks, can lead to situations where the technique will fail to yield the pit with the maximum value [Wright, 1990]. It does not solve the problem of overlapping blocks. This problem arises from the fact that if the blocks are considered separately with their individual cones, the sign of the cones may be negative. However, if such blocks are considered together with their cones combined, a positive-valued pit can be obtained.

In the 3-D positive moving cone technique, the search commences at the north-west corner (top left) of the grid block model and proceeds from west to east (left to right) along each level. All blocks with positive value on the first level are examined, before proceeding to the second level and then to the third level until the ultimate pit depth. Finally the 3-D block representation of the optimum pit is a combination of the optimum 2-D cross-sections and longitudinal sections. Smoothing is employed to fit the sections into 3-D pits after the optimum pit limits on 2-D sections have already been designed.

2.4.3 Lerchs-Grossmann's Optimum Open Pit Mine Design

Lerchs and Grossmann (1965) have used the graph theory and dynamic programming to formulate an open pit model to define the optimum pit limits. The objective of Lerchs-Grossmann's open pit model is to design the final pit limit of an open pit mine which maximizes the difference between the total mine value of ore extracted and the total extraction cost. The model is based on the following assumptions: (i) the type of minerals, its mine value and extraction cost are given for each point and (ii) the restrictions on the geometry are specified (surface boundaries and maximum allowable wall slopes).

Dynamic programming algorithm is used for the 2-D pit (or a single vertical section of a mine), and a graph theory algorithm for the general 3-D pit. Density functions of mine value of ore per unit volume, extraction cost per unit volume, and profit per unit volume are defined at each point of a 3-D space. In developing the 2-D pit model, the whole pit is divided into parallel and vertical sections and each section is considered as a 2-D pit. The technique used to determine the contour of a section consists of moving three straight lines which represent the bottom of the pit and two walls at the slope angles, and evaluating the ore and the extraction cost of materials limited by the three lines. The configuration of lines yielding the best results is then selected. Here, the angle is taken to be constant over the entire pit.

In the 3-D model using a graph algorithm, the entire pit is divided into a set of volumes defined by a 3-D grid. The Lerchs-Grossmann algorithm converts the 3-D grid of blocks in the orebody model into a directed graph. Each block in the grid is represented by a vertex which is assigned a mass equal to the net revenue value of the corresponding block. The vertices are connected by arcs (arrows) in such a way that the connections leading from a particular vertex to the surface define the set of vertices (blocks) which must be removed if that vertex (block) is to be mined. Lerchs-Grossmann's open pit optimization algorithm assumes a constant slope angle over the entire pit. In mining practice the pit slopes are determined by the rock mechanics characteristics and the hydrogeological conditions of the mine. These conditions may result in variable pit slopes all over the entire mine. In the 2-D model, the whole pit is divided into parallel vertical sections and each section is considered as a 2-D pit. When the optimum contour of all the vertical sections are assembled, it invariably turns out that they do not fit together. The resulting contour may be far from optimum. This undermines the reliability and credibility of the model. The problem with the 3-D model is that it presents a tight choice between a true optimizing algorithm characterized by large computing requirements and a non-optimizing algorithm with lower demands on computing facilities [Carlson, et al., 1966].

2.4.4 The Dynamic Programming Algorithm by Koenigsberg

Koenigsberg (1982) and Wilke and Wright (1984) have succeeded in applying dynamic programming directly to solve the 3-D pit design problem. In contrast to the 2-D case in which a given block can have nearest neighbours from only one column in the backward direction, in the 3-D case a block can have neighbours from four columns. The required slope angle, 1:1 in this case, must be satisfied with respect to each of the neighbouring columns. Figure 2.4 shows the 12 nearest neighbours to block b_{ijk} in the backward direction. With the neighbours in the backward direction identified, the pit value, P_{ijk}, on the block (i,j,k) is the sum of block column value for a given block b_{ijk} and the maximum pit value on a block among the neighbouring blocks. This is the optimum

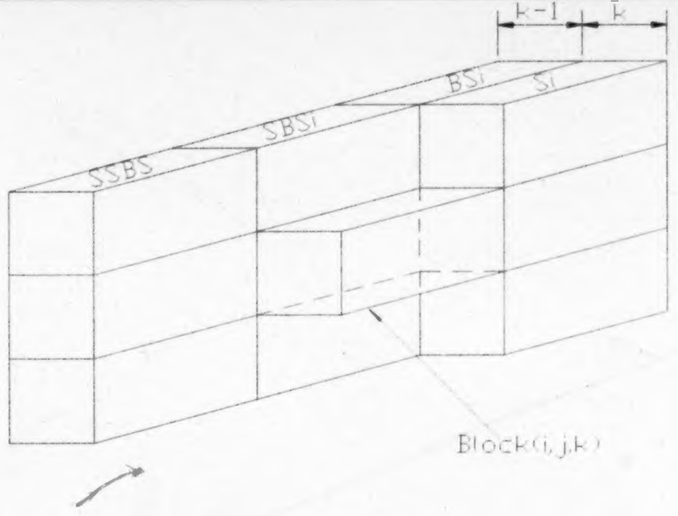


Figure 2.4 The 12 Nearest Neighbours to Block b_{ijk} in the Backward Direction.

value on the block b_{ijk}. The block column M_{ijk} for a given block b_{ijk} is computed in the same manner as for M_{ij} in the 2-D case. The other terms in the recursion formula are interpreted in a similar manner. The negative terms are corrections. These ensure that the neighbours eventually selected as the optimum neighbours are indeed compatible to each other, and to the block in question, with respect to the slope angle restrictions.

The problem with this algorithm is that it degenerates due to the use of a 2-D increment, M_{ijk}, for the progressive widening of the ultimate pit. This may lead to situations in which blocks selected from the four neighbouring columns end up being incompatible to each other in terms of slope requirements. Further, Shenggui and Starfield (1985) have shown that Koenigsberg algorithm is actually over constrained, and, in some cases, will actually miss the optimum ultimate pit. The problem of degeneration encountered with the 3-D algorithm by Koenigsberg was solved by Wilke and Wright (1984) with the use of 3-D increments in the form of minimum removal cones over each block.

2.4.5 3D Dynamic Programming (Wilke and Wright)

The problem of degeneration encountered with the 3-D algorithm by Koenigsberg was overcome by Wilke and Wright (1984) with the use of 3-D increments in the form of minimum removal cones over each block. This necessitates a new formulation of the recursion formula for the block pit value, P_{ijk}. However, the basic procedure of the 2-D dynamic programming case is preserved and remains valid; namely, the pit value, P_{ijk}, on the block, b_{ijk}, can be calculated from the value of the removal cone over and including the block b_{ijk}, added to the best value neighbouring pit compatible with the block b_{ijk}. It must, however, be ensured that no block value is counted more than once. Thus, all the block values, M_{ijk}, which are contained in both the removal cone, C_{ijk}, and in the value of the respective neighbouring pits $P_{i-1,\, j\pm 1,\, k}$, $P_{i,\, j\pm 1,\, k}$ and $P_{i+1,\, j\pm 1,\, k}$ must be deducted from the neighbouring pit values. According Wright (1990), the optimum value of the block *bijk* is given by the recurssive formula,

$$P_{ijk} = C_{ijk} + Max\begin{cases} P_{i-1,j\pm1,k} - P_{i-1,j\pm1,k} \cap C_{ijk} \\ P_{i,j\pm1,k} - P_{i,j\pm1,k} \cap C_{ijk} \\ P_{i+1,j\pm1,k} - P_{i-1,j\pm1,k} - C_{ijk} \end{cases} \qquad 2.19$$

2.4.6 Dynamic Path Level Pit Design and Optimization

The 3-D dynamic path level algorithm becomes a very useful approach when positive value blocks in the upper levels contribute to pit values in the lower levels. In this case, the dynamic path algorithm which employs a sectional pit optimization approach fails. Thus employing the dynamic path in level-wise instead of section-wise results in the correct answer. The steps involved in this approach are (i) the application of the dynamic path to the first level of the block model to determine and mine the value of the positive pit if one exists, (ii) apply the dynamic path to the remaining first and second levels and mine any positive valued pits, and (iii) add another deeper ievel, one at a time until final pit limits are reached (Wright, 1990).

2.4.7 Machine Learning Algorithms and Open Pit Mine Design and Optimization

It is apparent from the above discussions that various heuristic and mathematical algorithms have been applied to the open pit design and optimization problem. Most of the algorithms have short-comings and their applications will lead to sub-optimal pits. In view of the complexity of the open pit mine design and optimization problem, machine learning algorithms may offer the best methodology for their resolution.

2.4.8 Neural Network (MCS/MFNN) Algorithm for Open Pit Optimization

Frimpong and Achireko (1997) introduced a set of algorithms that address the random field properties of the various parameters involved in open pit optimization. This thesis is directed towards the development of a new stochastic algorithm to solve the open pit optimization problem. The modified conditional simulation and multilayer feed-forward error-back propagation algorithm was developed to capture the stochastic behavior of ore grades and reserves on one hand and to utilize neural network architecture in the optimization process. The modified conditional simulation (MCS) algorithm involves

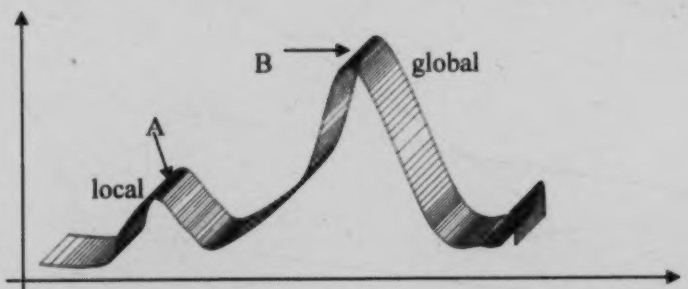


Figure 2.5 A Local and Global Maxima in A Typical Optimization Problem.

the use of linear average subdivision technique (LAS) and best linear unbiased estimation (BLUE) to model the stochastic nature of ore reserves and grade. In this case conditional simulation is employed to model the grade using the mean and the standard deviation. The MFNN is then used for classifying and partitioning the conditionally simulated blocks.

The 2-D model as presented by this algorithm does not represent reality. As depicted in the Figure 2.5, there is the possibility of the existence of both local and global maxima in a maximization problem. The learning algorithm can converge to a solution that does not minimize the mean square error. The back-propagation algorithm can converge to point A, a local maximum point (sub-optimum) with one set of initial conditions and converge to point B, a global maximum (optimum) with another set of initial conditions (Hagan, Demuth and Beale, 1995). To ensure that a global optimum point has been achieved, several different initial conditions have to be evaluated. The situation may even get worse in multi-dimensional problem with multiple parameters. Therefore an optimal solution may not be achieved when a back-propagation algorithm is used to solve this problem. Therefore there is the need to search for a robust technique that can overcome the limitations of the back-propagation algorithm. The above literature review is indicative of the various attempts at solving the open pit design and optimization.

2.5 Overview of Machine Learning

Machine learning can be described as the application of artificial intelligence, probability and statistics, computational complexity theory, control theory, information theory, philosophy, psychology and neurobiology to the solution of computationally intractable problems. Machine learning involves searching through a hypothesis space defined by an underlying representation (artificial neural network, decision trees, linear functions, logical descriptions, and fuzzy logic) to determine the hypothesis that best fits the given data and any prior constraints or knowledge [Mitchell, 1997]. As opposed to natural learning, machine learning is essentially the use of computer programs based on some algorithm (hypothesis) to improve their performance of some task through experience.

2.5.1 Neural Network

The human audio/visual capabilities are superior to any kind of super-computer in mankind's history. The brain is composed of a large number, (10^{11}), of nonlinear processing unit called neurons. Figure 2.6 depicts a biological neuron reported in the literature [Haykin, 1994; Hertz et al, 1991; Zurada, 1992]. The dendrites are nerve cells that are connected to the neuron body, called the soma. Axons are the long fiber cables extending from the soma. Synapses are located at the end of axons, and are connected to other neurons. An axon is composed of several thousand synapses. Information transmission from one neuron to another takes place at a synapse which is considered to be a complicated chemical process.

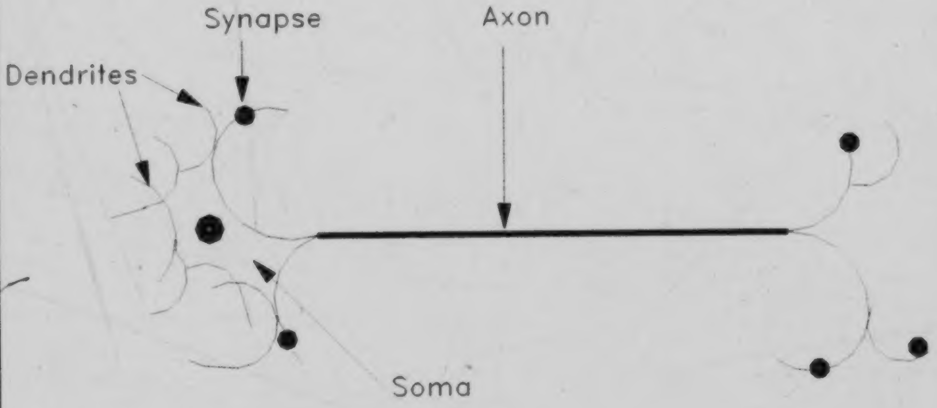


Figure 2.6 A Biological Neuron.

The goal is to increase or decrease the potential of a neuron that is receiving information. If the potential exceeds a threshold limit set in the receiving neuron, the neuron is triggered to send an information along the axon to the other neurons. The way these neurons are connected to each other makes us capable of recognizing speech or images very easily. Biological neurons are robust and fault tolerant. This means that if one or more neurons die or do not function properly, the overall performance of the brain does not change. They are highly parallel, very small, compact and low powered and they can deal with noisy, inconsistent data with no difficulty. These features motivated the new research field called artificial neural networks. McCulloch and Pitts (1943), in their initial attempt at modeling a neuron, proposed a binary threshold unit as a neuron. Their model is depicted in Figure 2.7 and equations 2.1 and 2.2. The McCulloch-Pitts neuron has proven to be capable of doing universal approximation for appropriately chosen weights as long as a number of them are connected in parallel and working simultaneously.

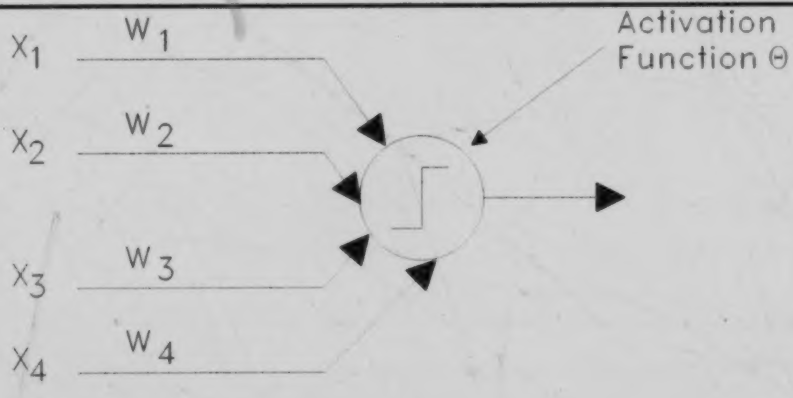


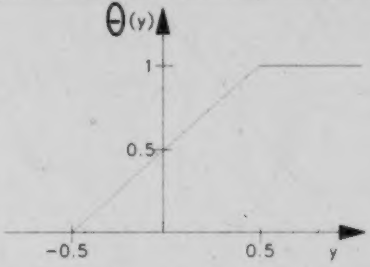
Figure 2.7 McCulloch-Pitts Neuron Model

$$\Theta(y) = \begin{cases} 1, & if \quad y \geq 0 \\ 0, & otherwise \end{cases} \qquad 2.20$$

$$y = \Theta\left(\sum_{i=0}^{M} w_i x_i\right) \qquad 2.21$$

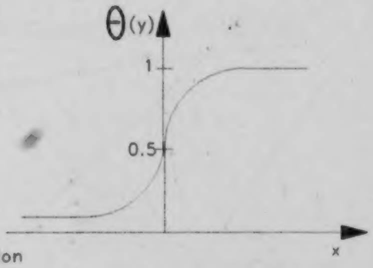
Haykin (1994) has described various activation functions. These activation functions as well as their mathematical representations are shown in Figure 2.8. The first one is piece wise function, and the latter one is sigmoidal function. Both logistic sigmoid function and hyperbolic tangent function are considered as sigmoidal functions.

$$\theta(y) = \begin{cases} 1, \text{ if } y > 1/2 \\ y, \text{ if } -1/2 > y > 1/2 \\ 0, \text{ if } y < -1/2 \end{cases}$$



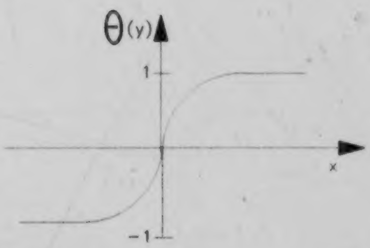
(a) Piece Wise Activation Function

$$\theta(y) = \frac{1}{1+e^{-x}}$$



(b) Logistic Sigmoidal Activation Function

$$\theta(y) = \frac{1+e^{-2x}}{1+e^{-2x}}$$



(c) Tanh(x) Sigmoidal Activation Function

Figure 2.8 Mathematical Representation of Activation Functions.

There are various types of neural network architectures. The famous neural network architectures are multilayer feedforward neural network (MFNN) or multi-layer perceptron, Hopfield network and Kohonen network. Each of them is described in the next sections.

2.5.2 Multilayer Feedforward Neural Network

Multilayer feedforward neural network consists of simple McCullogh-Pitts type neurons in a massively parallel structure (Figure 2.9). It is composed of an input layer, also called sensory unit layer, hidden layer, which could be as many as necessary, and an output layer. The input signal is usually propagated forward layer by layer. These types of artificial neural networks are referred to as multilayer feedforward (perceptrons). These networks were widely used to solve some classification and approximation problems employing a very popular training algorithm named error back propagation (EBP). The EBP is a learning scheme where the error is backpropagated and used to update the weights.

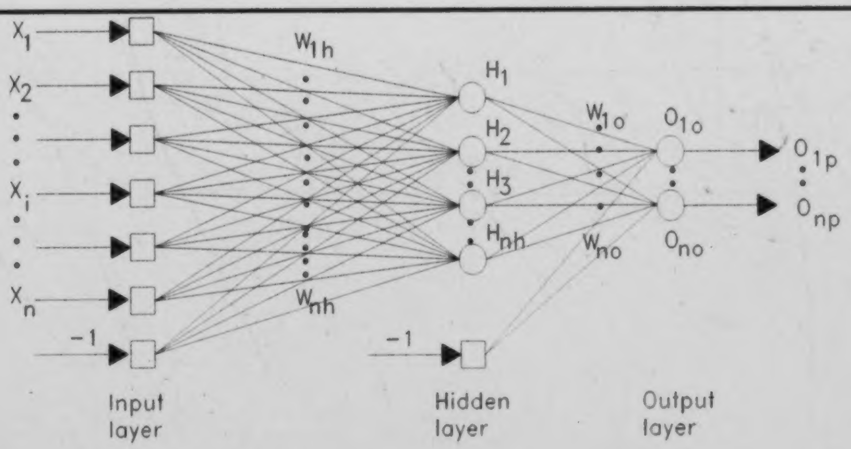


Figure 2.9 Multilayer Feedforward Neural Network

In a multilayer feedforward neural network (Figure 2.9) the nodes are grouped in input, hidden, and output layers, by the network. The result of the computation is displayed by the output layer. The hidden or intermediate layers follow the input layer. Generally all the nodes within the input layer are connected to all the nodes of the first hidden layer which are subsequently connected to all the nodes of the second and other hidden layers and eventually to the nodes of the output layer. A weight, depicted in Figure 2.9, is associated with each one of the connections. The -1's in Figure 2.9 are biases to the input and hidden layers.

The input nodes perceive the data to be processed by the network and then transmit the results to the hidden nodes. Within the hidden layers, the values received from the input layer nodes are multiplied by the connection weights and then each hidden node sums all the weighted values it receives. On the other hand any other type of linear transformation can be used instead of the usual arithmetic summation. Finally a non-linear transformation, such as a sigmoid function, is applied to the summation result and the resulting value (the activation level of the node) is then transferred to each of the nodes of the second hidden layer or of the output layer. The output layer plays the same role as the hidden layer however, the linear and non-linear transfer functions within the output layer can be different from those of the hidden layer. The activation level of the output nodes are the network outputs or results.

2.5.3 Hopfield Network

The Hopfield network consists of two layers, an input layer and a Hopfield layer (Figure 2.10). Each node in the input layer is directly connected to only Hopfield layer. The nodes in the latter layer are neuron models previously described with either hard limiting or sigmoidal activation functions [Hopfield, 1982]. The outputs of these nodes are weighted and fed back to the inputs of all of the other nodes.

During training, the network output is often required to be the same as the input. Connection strengths are weakened by reducing the corresponding weight values if the output of a neuron is different from the input, and strengthened when the converse is true. The trained network is used by applying an input pattern to the network. The network outputs are then continually fed back through the weights until a convergence criterion is met, typically when there are no changes at the network output nodes on successive iterations.

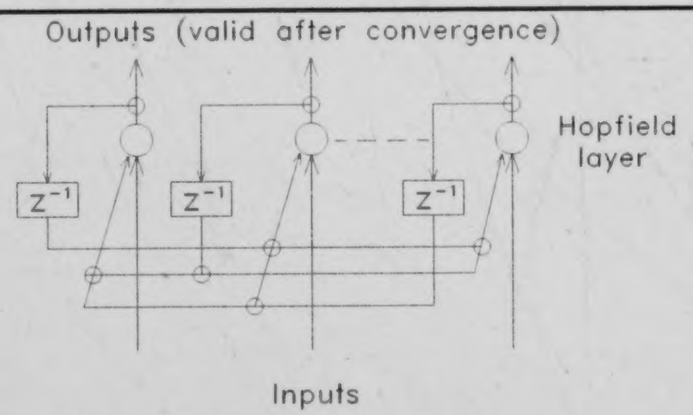


Figure 2.10 Hopfield Network. z^{-1} is the Unit Delay Operator.

This is the final network output for the input pattern. Binary input and output values, often represented as + 1 and - 1, are usually used with the Hopfield network.

2.5.4 Kohonen Network

The main distinguishing feature of this network, from the MFNN and Hopfield networks, is that no output data is required for training [Gomm, Page and Williams, 1993]. A Kohonen network is constructed with a fully interconnected array of neurons (i.e. the output of each neuron is an input to all neurons, including itself) and each neuron receives the input pattern (Figure 2.11) [Kohonen, 1988]. There are two sets of weights: an adaptable set to compute the weighted sum of the external inputs and a fixed set between neurons that controls neuron interactions in the network.

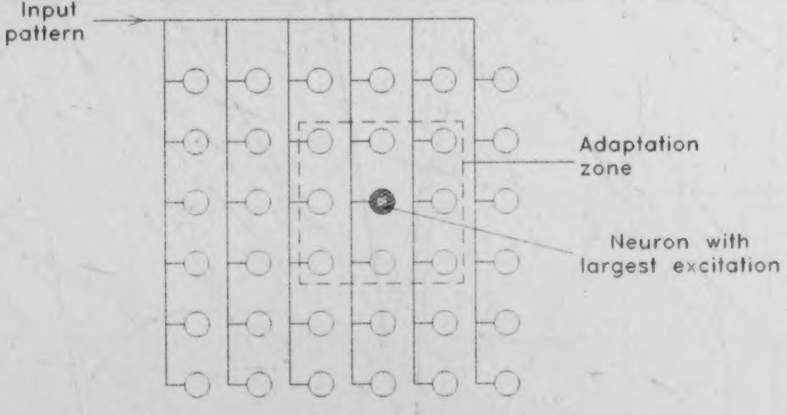


Figure 2.11 Kohonen Network

Training involves applying an input pattern to the network, consisting of a set of continuous-valued data, and the output of each neuron is computed. The neurons are then allowed to interact with each other and the neuron that responds most to the input stimuli (i.e. the one that has the largest output) is found.

Only this neuron and neighbourhood neurons, within a certain distance, are allowed to adjust their weights to become more responsive to the particular input. This form of training has the effect of organizing the "map" of the output nodes such that different areas of the map will respond to different input patterns. Hence, the Kohonen network has self-organizing properties and is capable of recognition.

2.6 Conclusion

In summary the various algorithms employed in the design and optimization of open pit mines have been inadequate. One of the main reasons for the failure of the various algorithms is the complexity of the design and optimization problem. The open pit mine design and optimization problem belongs to the class of complex problems that can not be solved by heuristic and traditional mathematical models. The use of mathematical modelling by machine learning, namely adaptive logic and neuro-genetic networks may resolve the problems associated with the foregone algorithms.

CHAPTER 3

THEORY AND PHILOSOPHY OF OPEN PIT DESIGN AND OPTIMIZATION

In this chapter, the philosophy, conceptual framework and theory of open pit design and optimization as presented in this thesis are discussed.

3.1 Design Paradigms and Optimality Criteria

Generally, the design and optimization of an open pit commence with a conception of the various objectives, goals and functions. On the basis of the requirements of the mine operator, the operational constraints and assumptions, a mine is designed to achieve the desired intentions. Different techniques are used to resolve various aspects of the design process. Using some criteria such as maximization of net value or net present value and minimization of costs, the layout of the designed open pit mine is optimized.

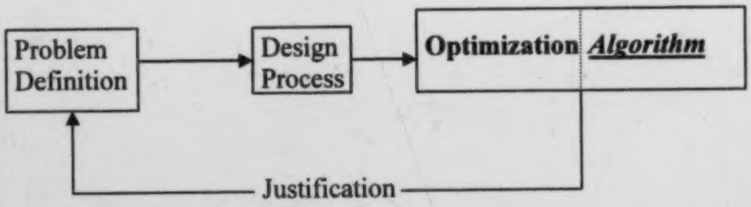
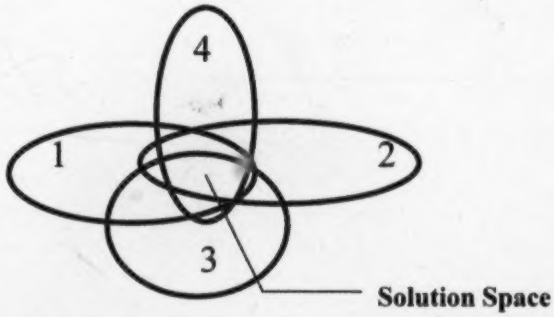


Figure 3.1 Generalized Design and Optimization Model

The success of the optimization process is dependent on the algorithm employed. The optimization algorithm is coded, run and the results are compared with other algorithms (and the problem definition (see Figure 3.1)). The design of an open pit optimization problem incorporates various processes, theories and/or hypotheses. In this work, the optimization problem is regarded as a physical system constrained by four major constraints: geotechnical, production costs/expenses, ore reserves/grade, and price as depicted in Figure 3.2.



Constraint 1 = geotechnical; Constraint 2 = cost; Constraint 3 = price; Constraint 4 = grade/block values.

Figure 3.2 Solution for a Typical Constrained Design.

The optimized solution or design is the solution space formed by the intersection of the four sets. In an over-constrained situation none of the sets would intersect, whereas the area of the solution space could be wider in an under-constrained solution. Both of these situations will not lead to an optimized solution and may require the relaxation or tightening up of the requirements or parameters. As depicted in Figure 3.3, the solution space is bounded by a lower and upper limits, UPL_1 and UPL_2 respectively. In the case of UPL_1, more ore can be taken for the same amount of waste already taken (increasing returns to scale). Whereas in UPL$_2$, for each ore mined a higher quantity of waste must be taken.

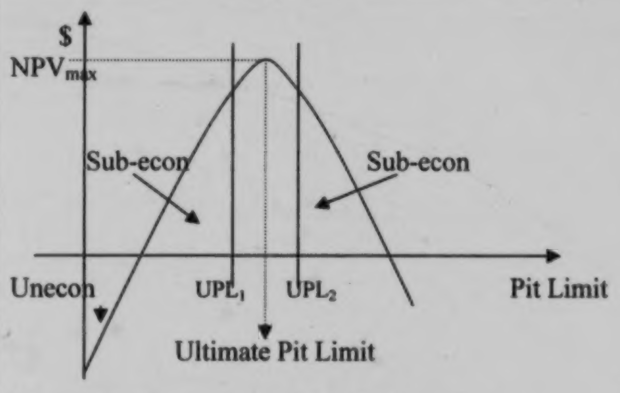


Figure 3.3 The Domain of the Solution Space

A higher discount rate may also have the same effects as a higher quantity of waste. Assuming there is perfect information, the optimized pit (the ultimate pit limit) is characterized by a maximum net present value, NPV$_{max}$. In reality (in the absence of perfect information), the relationship between the optimized pit and NPV$_{max}$ is uncertain and can only be captured by a stochastic model. An efficient optimization algorithm is the one that can search the solution space till it finds the optimum pit limit. Most of the optimization (search) algorithms, especially the learning algorithms approach the optimum pit limit assymptotically as shown is Figure 3.4. An inferior algorithm may take a longer time to reach the optimum, whereas an efficient algorithm will reach the optimum at a faster rate.

Another interesting feature of an inferior algorithm is that it may not reach the optimum at all, independent of the amount of time it is run. An algorithm may stabilize as it gets near the optimum and may never get to the true optimum resulting in a degree of error xy.

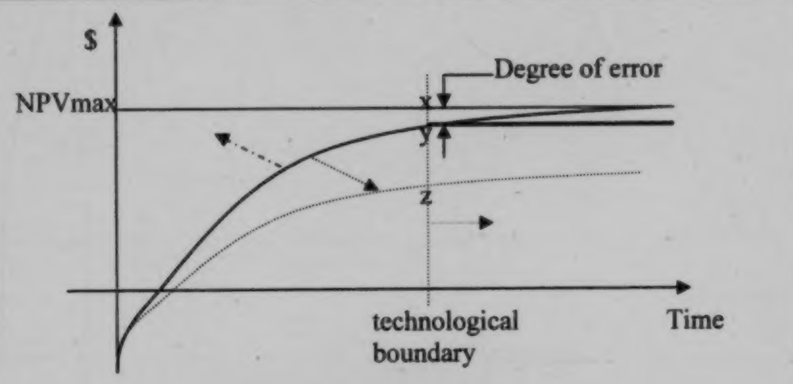


Figure 3.4 The Optimization Process.

As depicted in Figure 3.4, if the algorithm is much more inferior, the degree of error is wider (xz). An inferior algorithm takes a longer time to solve the problem, consumes more computer memory. Algorithms which searches only the sub-economic sub-spaces can also be considered as inferior since the quality of their final answers are sub-optimal.

3.1.1 Design Evaluation

If the state of the design solution is evaluated at regular intervals or stages of the design process, chances are that the resulting design (end product) may be optimal and satisfy the desired constraints/criteria. Such an analysis can be done employing an amalgamation of probability, statistics, machine learning and quality methods/techniques (Green, 1997). It is assumed among other things that the design space can be subdivided into subspaces. The benefits of the design process is maximized by employing advanced technology and a first-right approach is adopted within the design process to minimize the number of deficiencies discovered during the optimization stages. In the mine design model described above, the design space is subdivided into sub-spaces or modules, each having a design characteristic (DC).

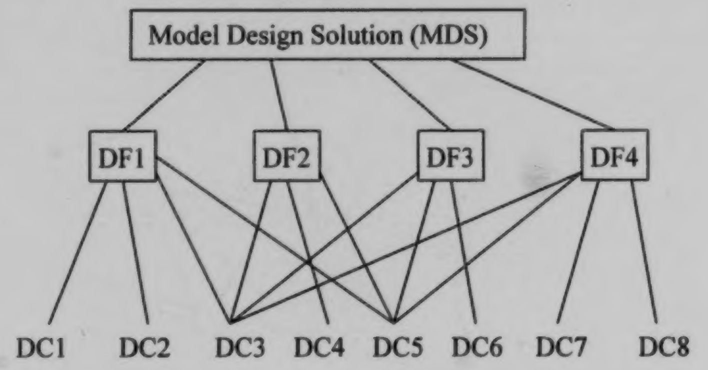


Figure 3.5: A Series Combination of Design Factors (After Green 1997)

The solution to the mine design (MDS) problem comprises of a set of design characteristics (DC's). Thus the DC's are the dimensions of the solution space which are subdivided into a finite number of cells each containing a potential solution model. A particular design characteristic is considered critical and denoted DCC, if it must be fully satisfied before a design process can proceed. If a particular universal set of design characteristics must be satisfied before the resulting design is accepted, then they can be modelled as a series system (see Figure 3.5). A union of a number of design characteristics is termed a design factor (DF). Generally in a complex design situation such as ore reserve modelling, a number of sub-models and design factors exists in a hierarchical manner. This hierarchy of interactive design levels combine to form a model design solution (MDS). In some situations, especially where there is less complexity a MDS may be a glorified DF. The impact of a particular characteristic can be defined in terms of the degree of match the target level of the design specification.

3.1.2 Design Uncertainty

Assume that each design characteristic has a variable target function $f(PDF)_T$ with minimum and maximum limits labeled $\min_T$ and $\max_T$ respectively. The corresponding function, minimum and maximum for the estimated value of the design characteristic are $f(PDF)_E$, $\min_E$ and $\max_E$, respectively. A probability density function can be employed to define the uncertainty associated with a particular design characteristic. At the early stages of a design where there is lack of information, a higher level of uncertainty can be associated with the design. According to Green(1997), the probability that the design estimate will fall within the desired limits are given by;

$$\int_{\max(\min_T,\min_E)}^{\min(\max_T,\max_E)} (PDF_E)du \qquad 3.1$$

In the presence of further information, the characteristic probability distribution function can be modelled. The degree of separation or overlap of the design characteristic target distribution $[f(PDF)_T]$ and design characteristic estimate distribution $[f(PDF)_E]$ is depicted in Figure 3.6.

The reliability domain describes the degree to which the design characteristic target and the design characteristic functions overlap. The reliability domain approaches zero as the degree of matching increases.

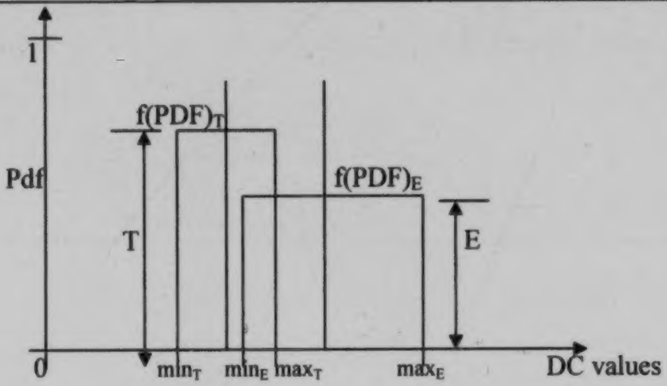

1
f(PDF)T
Pdf
f(PDF)E
T
E
0
minT
minE
maxT
maxE
DC values

Figure 3.6 Design Characteristic Values with a Uniform Distribution

(After Green, 1997)

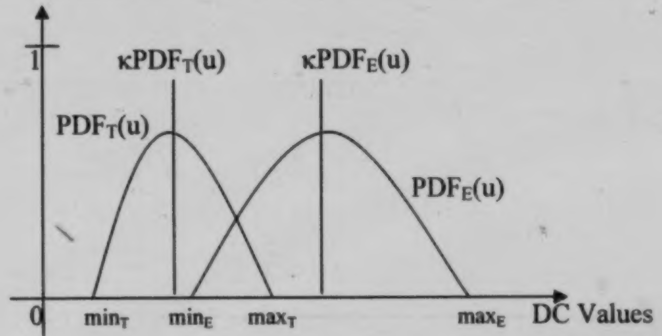

1
κPDFT(u)
κPDFE(u)
PDFT(u)
PDFE(u)
0
minT
minE
maxT
maxE
DC Values

Figure 3.7 Design Characteristic Values with a Non-uniform Distribution

(After Green, 1997)

The reliability domain or design margin is given by (Green 1997),

$$RD = \frac{\kappa f(PDF_T) - \kappa(PDF_E)}{\sqrt{\left(\sigma_T^2 + \sigma_E^2\right)}} \qquad 3.2$$

Where κPDF$_T$(u) and κPDF$_E$(u) are the mean target distribution and estimate distribution values respectively.

3.1.3 Interaction of Design Characteristics

The joint probability that the estimate and the target values fall within each other's limits is given by Green(1997) as,

$$\prod_{i=1}^{n}\left[\int_{\max(\min_T, \min_E)}^{\min(\max_T, \max_E)} PDF_E(u)du\right] \rightarrow 0 \qquad 3.3$$

In a situation where the design characteristics are noncritical and can therefore be modelled in parallel, the probability that the target values will not be met is given by Green (1997) as,

$$\prod_{i=1}^{n}\left[1 - \int_{\max(\min_T, \min_E)}^{\min(\max_T, \max_E)} PDF_E(u)du\right] \rightarrow 0 \qquad 3.4$$

3.1.4 Optimization and Optimality Criteria

The optimization process is outlined in the lower part of Figures 3.8 and 3.9 below. There are two types of inputs into the model, namely soft data or domain knowledge and hard data. The hard data consist of grade values, price, costs and rock properties. The domain knowledge is basically qualitative hints.

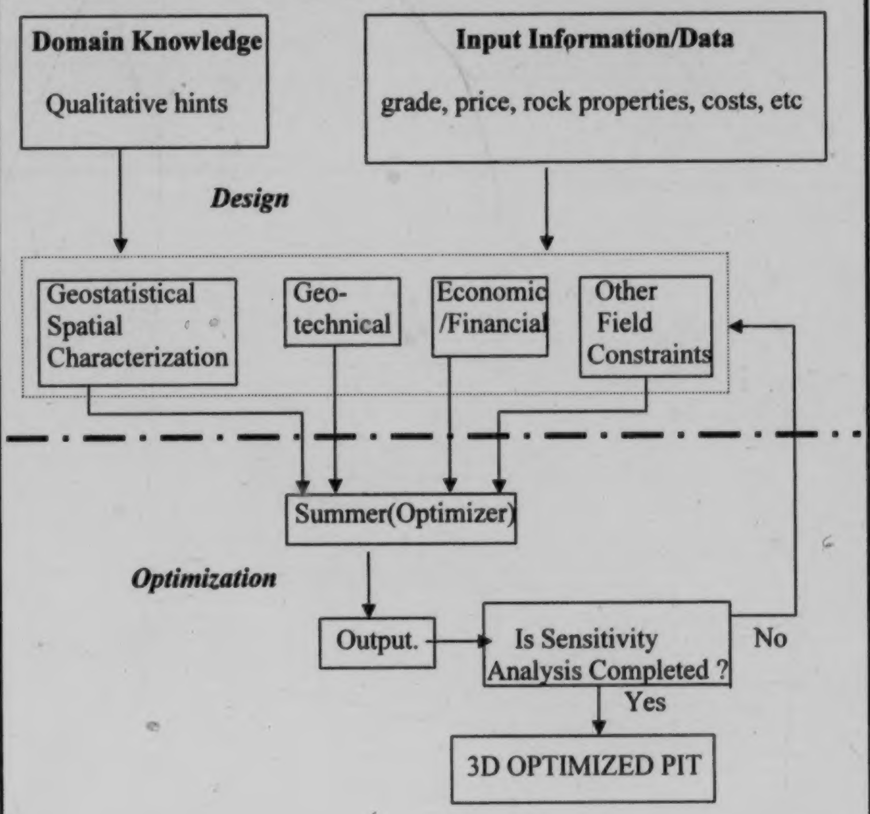

Domain Knowledge
Qualitative hints
Input Information/Data
grade, price, rock properties, costs, etc
Design
Geostatistical Spatial Characterization
Geo-technical
Economic /Financial
Other Field Constraints
Summer(Optimizer)
Optimization
Output.
Is Sensitivity Analysis Completed ?
No
Yes
3D OPTIMIZED PIT

Figure 3.8: A Traditional Open Pit Design and Optimization Process

These data are fed into the corresponding parts of the model (spatial characterization, geotechnical, economic and other field constraints sub-models). As depicted in Figure 3.8, the output from the first stage of the design phase of the modelling become the input of the summer(optimizer), which is the machine learning algorithm. The results of the optimization algorithm is subjected to sensitivity analysis to yield the final 3-D optimized pit.

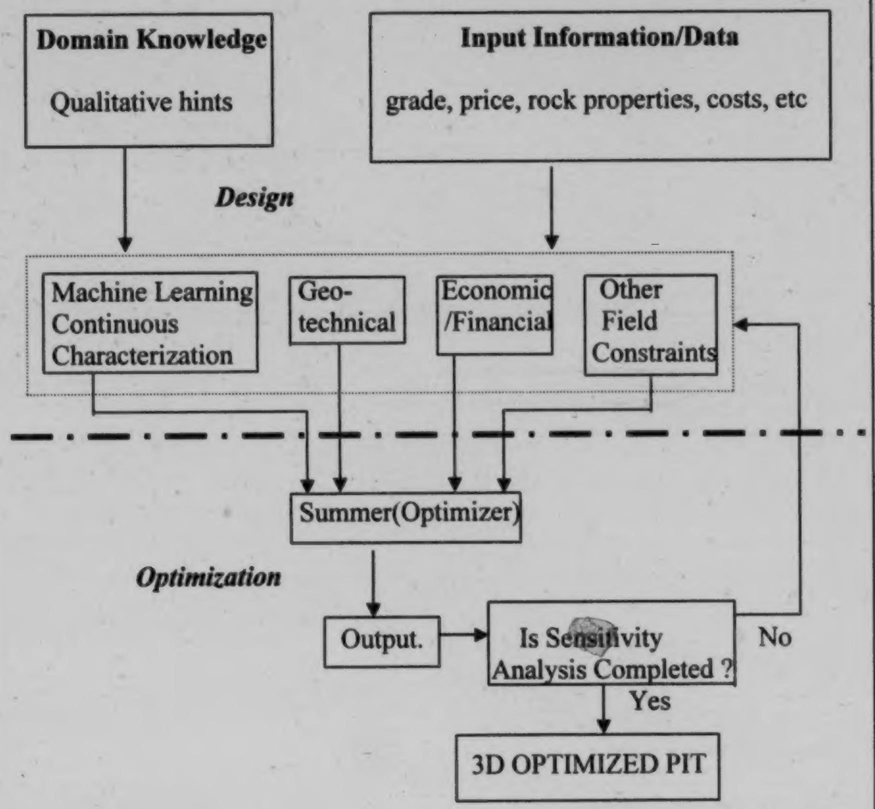

Domain Knowledge
Qualitative hints
Input Information/Data
grade, price, rock properties, costs, etc
Design
Machine Learning Continuous Characterization
Geo-technical
Economic /Financial
Other Field Constraints
Summer(Optimizer)
Optimization
Output.
Is Sensitivity Analysis Completed ?
No
Yes
3D OPTIMIZED PIT

Figure 3.9: An Intelligent Open Pit Design and Optimization Process

3.2 Computer Modelling of Mineral Deposits

There are two broad approaches to building 3-D computer models of orebodies, namely geometrical and volumetric modelling.

Geometrical modelling techniques (regular block, gridded seam, serial slice, solid geometry, mathematical functions and boundary representations) are geared towards the identification and interpretation of orebodies with well-defined boundaries. The volume of the geologically different materials in these boundaries are then estimated. A summary of the geometric modelling structures applicable to mineral deposits is presented in Table 3.1 below. Volumetric modelling of ore deposits is used to deal with attributes that vary continuously within the volumes defined by geometric modelling.

Table 3.1 Geometrical Modelling Techniques for Orebodies

Model Type	**Description**
Block	A series of 3-D square/rectangular blocks are used to represent the volume being modelled.
Serial-Slice	A number of polygonal shapes are digitized from or interpreted on series of regularly-spaced parallel sections or plans representing parts of the mineralized orebody. Polygons are assigned a volume of influence halfway to the next section.
Gridded Seam Block	Grids of elevation and/or thicknesses are used to represent multiple sub-parallel surfaces.
Solid	Groups of primitives(e.g. planes, cubes, spheres, etc.) are used to represent complex objects and thus define volumes with common properties.
Mathematical	Mathematical functions (polynomial trend surfaces, cubic B-splines, etc.) are used to represent discontinuities in a global or piecewise fashion.
Boundary Representation	A collection of points (vertices) and planar surfaces (facets) are used to represent volume boundaries.

Such properties are normally defined through sampling and geostatisical modelling. After defining the volumes of the main geological units, geostatistical techniques are then applied to define ore reserve block grade estimates taking into account block size, position, orientation and spatial distribution of the sample values.

3.2.1 Block Modelling

A 3-D block model of the mineralization is fundamental to the implementation of any open pit optimization algorithm. In open pit optimization, the depth and width of the pit from which the ore is extracted at a profit is defined using an open pit optimization algorithm. The first step involves building a block model and assigning grades to each block in the model using a database of grade samples from drill-holes. An interpolation technique is then used to assign grades to all the blocks. The maximum allowable slope based on rock mechanics studies of the ore deposit and the wall rock is factored into the optimization algorithm.

It cannot be gainsaid that an accurate and reliable ore reserve estimate is fundamental to the success of any mining operation (Lane, 1997). Reserve risk is an important component of the mine feasibility, optimization and operations process. A banker looks for such key aspects as ore model construction, lithology, compositing, data analysis and statistics, variography, interpolation, recovery and dilution factors, and reserve estimates. The success of any optimization algorithm is dependent on estimates of reserves and grade together with the assessment of uncertainty and risk. In this study geostatistical modelling is used to provide reserve estimates and the associated risks.

Geostatistical study of a mineralized zone of interest is a multiple-stage process (see Figure 3.10). The first stage of an actual geostatistical study involves the modelling of the spatial variability within the study area. The variogram can be simply defined as the relationship between geological distance and Euclidean distance. The variogram quantifies the spatial variability of the contained mineral/metal by computing the variance of the grade values measured some distance h, apart. The variogram/variance increases with the separation distance. Kriging techniques are estimation methods that minimize the error variance of estimation (as calculated with the variogram model) for

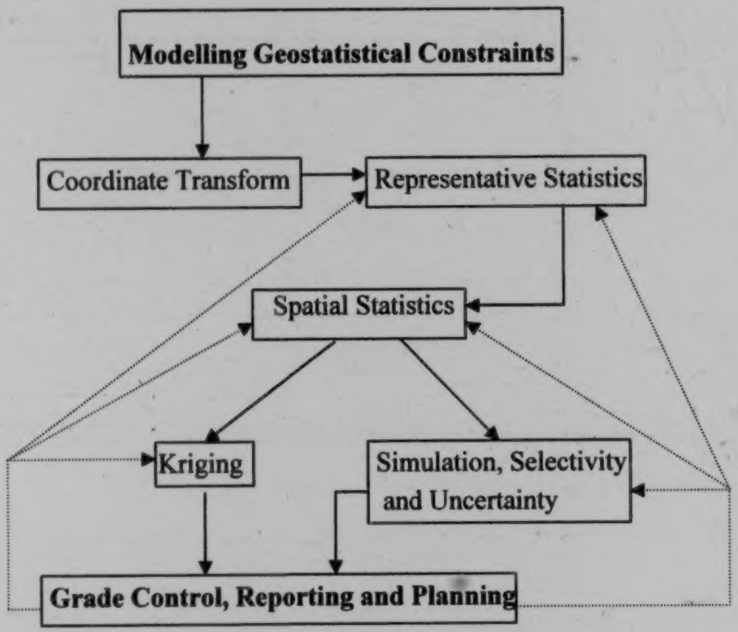

Modelling Geostatistical Constraints
Coordinate Transform
Representative Statistics
Spatial Statistics
Kriging
Simulation, Selectivity and Uncertainty
Grade Control, Reporting and Planning

Figure 3.10: Sequence of Actions in Geostatistical Modelling and Evaluation of Ore Reserves.

spatially distributed data. The kriging estimation variance can be affected by errors in the calculation of the variogram.

Some of the reasons for estimation errors include (1) sparse data, (2) high variability, and (3) incorrect assumptions in the modelling procedure. Kriging-type estimates are known to be smooth (that is they have less variability than the original data) and can not therefore be used directly for mine planning. Simulation of the mineral/metal values will be employed to correct for this smoothing effect. One major advantage of simulation is that the simulation model corrects for smoothing regardless of the data spacing and provides a measure of uncertainty about the predicted grades. The level of statistical, geostatistical and systematic uncertainty (e.g. head grade, tonnage,) as well as the risk (probability of loss or gain) associated with the numbers are modeled in order to calculate at some degree of confidence the global reserves and short term mine plans. This enables the decision-maker to quantify the degree of uncertainty, the level of risk and economic return associated with the optimized pit value obtained.

The results of the geostatistical analysis will be the estimation of resources/reserves and classification into various categories. It is evident that the entire in-situ mineral resource identified and estimated by geostatistical analysis cannot be mined because of variable mining and treatment costs and spatial variability inherent in the deposit. Besides geological insitu resources, mineable or recoverable reserves according to Journel and Huijbregts (1978) depend on such factors as selection criteria, cut-off grade or minimum mineable thickness, technological constraints, support effects, information available at the time of selection, geological boundaries and regulatory requirements. The influence of support and level of information is expressed by the geostatistical variances of dispersion and estimation.

The stochastic gold price as modelled via multiple regression and multilayer feed forward neural networks (Achireko and Frimpong, 1996) are employed in this work.

3.2.2 Block Model for Open Pit Design

A block can be defined as the smallest basic volume of material to which it is practical to assign grade, tonnage and geologic variability at an acceptable level of dilution. The block model is the basis for almost all computer supported pit designs. For a block model, an imaginary rectangular block large enough to cover the area of interest is placed around the mineral deposit [Wright, 1990]. The large block, as shown in Figure 3.11, is then subdivided into smaller three-dimensional blocks (Figure 3.12). The smaller blocks may be of various different sizes and shapes. The geometrical position of a block is uniquely fixed with reference to any suitable coordinate system. Each block is assigned geological, rock mechanical, processing and economic data pertaining to each type of material contained in the block. Though several types of block models exist (Kim, 1978), the regular 3-D fixed block model is the most widely used one in practice. The vertical height of each block is usually the bench height and the horizontal shape will normally be a rectangle or a square. The main characteristic of a 3-D fixed block model is that each block has the same measurements.

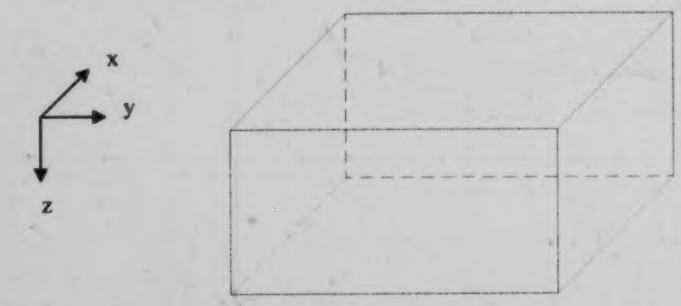


Figure 3.11 Large rectangular box to encompass area of interest

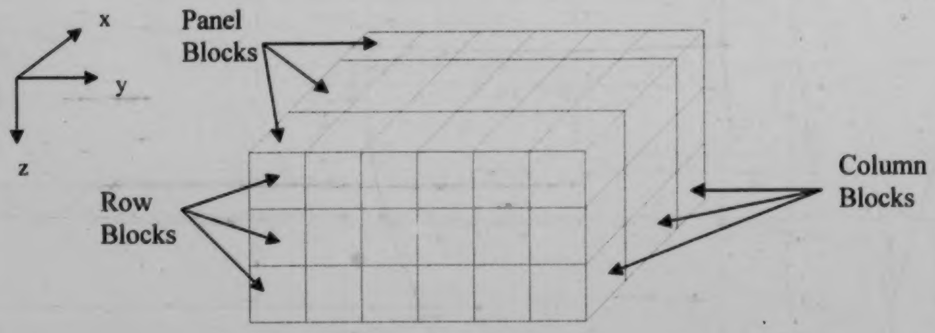


Figure 3.12 Large box is subdivided into smaller three-dimensional blocks

The size of a block are dependent on such factors as grade variability, geologic continuity, machine-time capabilities, slope stability and data storage limits. Though geostatistics is the dominant method for determining the minimum block size, the presence of major discontinuities throughout the deposit affect the block model. By employing such numerical techniques as geostatistics (kriging and simulation), inverse distance weighting methods and polygonal method, data are assigned to each block from a given database. Geostatistical simulation is superior to such reserve estimation methods as triangulation, inverse distance and polygonal. This is because simulation honors the mean and is more likely to result in an accurate representation of the deposit.

3.3 The Economic Value of a Block

The criterion for maximizing the pit value is equivalent to finding the collection of blocks within a pit limit within the layout, which will give the maximum possible value; subject to mine stability and mining constraints. Hence the economic value of a block is of utmost importance. The economic value of a block value (EBV) is defined by the equation;

$$EBV = I - DC - IC \qquad 3.5$$

Ore blocks and blocks containing both ore and waste (= mixed blocks) will have EBV less than zero, equal to zero or greater than zero, depending on the amount and quality of the ore contained in such blocks. Waste blocks will always have a negative EBV since income from waste is zero. The optimization criterion for the pit limit design problem can thus be stated as:

$$Maximize\ Z = \sum(EBV)_j \qquad 3.6$$

subject to slope stability and mining constraints.

After the determination of the EBV, the algorithm for combining the blocks which will yield the maximum profit is applied to the gridded blocks. A comprehensive treatment of this section is accorded in Chapter 4. In Chapter 4 the principles underlying the algorithm used for searching for the optimized pit layout is also discussed.

3.4 Rock Slope Design

Stochastic processes have been known to underlie rock formation. The use of discrete factor of safety values are therefore not reliable. The probabilistic approaches to open pit slope design have been advanced by such workers as Pine (1998), Priest and Brown (1983), Coates (1977), McMahon (1975), Piteau and Martin (1977) and Moss and Steffen (1978). The measure that will be employed in this work is the reliability index, which considers both the mean factor of safety and its variability.

3.4.1 Slope Design

In classical slope design the factor of safety is considered as;

$$F.S = \frac{Resisting \quad Forces}{Disturbing \quad Forces} = \frac{Strength}{Stress} = \frac{R}{D} \qquad 3.7$$

For a plane slope failure mode with a unit thickness, the factor of safety is given by

$$F.S = \frac{\left\{c * A + \left\lfloor W * \cos\psi_p - U - V * \sin\psi_p \right\rfloor \tan\Phi\right\}}{\left\{W * \cos\psi_p - U - V * \sin\psi_p\right\}} \qquad 3.8$$

The slope is considered to have failed if the safety factor < 1. Though there are many slope failures the plane failure mode is used in this work.

3.4.2 Uncertainty Analysis

In geomechanical design, there are 4 approaches which can be employed to deal with uncertainty in the factor of safety and the risk associated with it. The first approach involves plotting the probability density functions (PDFs) of all the variables in equation 3.8. The factor of safety is calculated from values which are randomly-selected from the various PDFs. Generally, Monte Carlo simulation is used to generate each of the variable values. The resulting cumulative density function (CDF) is used to calculate the coefficient of reliability or the coefficient of failure and the consequences of failure. The second approach used in dealing with uncertainty and risk is the safety margin which is defined here as the difference between the resisting forces and the disturbing forces, equation 3.9. In this case failure occurs if S.M=0.

$$S.M = \left\{c * A + \left\lfloor W * \cos\psi_p - U - V * \sin\psi_p \right\rfloor \tan\Phi\right\} - \left\{W * \cos\psi_p - U - V * \sin\psi_p\right\} \qquad 3.9$$

The reliability index for n units, with the probability of failure, p_n is given by

$$r_n = (r_1)^n \qquad 3.10$$

Where r_1 if the reliability of a single unit. Note that the reliability index of the ith cell of unit, r_i is given by;

$$r_i = 1 - p_i \tag{3.11}$$

The input variables and the safety margin are analyzed statistically. In this case Latin hypercube sampling together with @Risk/Excel can be employed to complete the calculation. The third approach involves geostatistical modelling and techniques. The fourth approach (which is the method employed in this work) involves the use of machine learning to develop the geomechanical design.

3.5 Optimization of Engineering Systems

Engineering optimization problems/systems can be classified into linear, non-linear and complex systems (Figure 3.13).

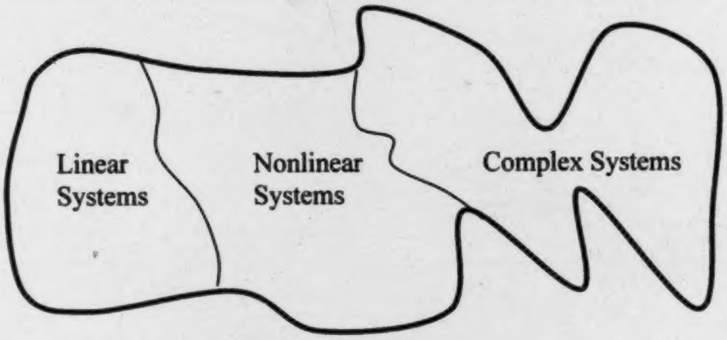


Figure 3.13 Classification of Engineering Systems

Linear systems are well-defined and have established optimization methodologies. Nonlinear systems are difficult to optimize and can be computationally tractable. Complex systems are computationally intractable and cannot be adequately modelled with mathematics. Even though it is possible to model and optimize nonlinear systems, their solution presents considerable difficulties. Nonlinear systems do not have closed form solutions and cannot therefore be solved analytically. Thus the process of solving non-linear optimization problems involves the use of numerical techniques to search for an optimal solution in the corresponding search space. It is quite difficult to obtain a global optimum for a nonlinear optimization problem. Using nonlinear objective functions entails dealing with a solution space with many local optima which makes the global optimum difficult to find. The nonlinear constraints can also form feasible regions that are hard to find and mostly difficult to deal with. Since the nonlinear constraints result in small feasible regions that are difficult to locate or very few feasible solutions (multi-modal problems), numerical search methods must be used with care.

In both constrained and unconstrained optimization problems, nonlinear multi-modal objective functions can be the most difficult to solve. Some of these challenges are illustrated in Figure 3.14. The terrain represents various characteristics that can cause problems for search methods that adapt to a subset of these characteristics (Shang, 1997). For instance, gradient search methods cannot effectively deal with the steep and gentle slopes at the same time. The large shallow basins and plateaus may equally be difficult to model as small step size may take a considerable amount of time to search.

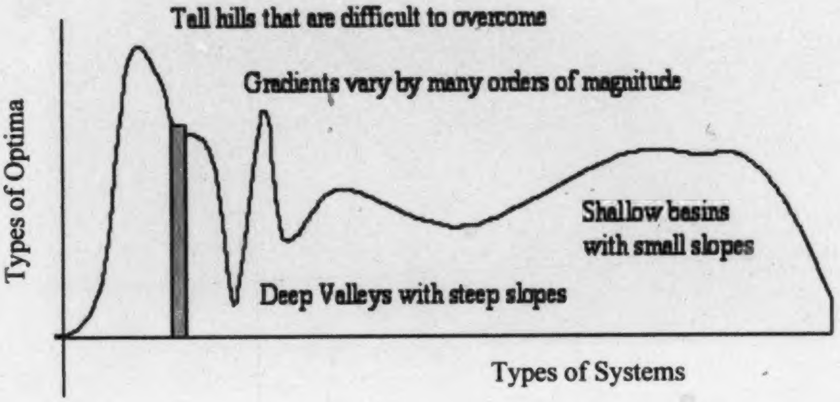

Tall hills that are difficult to overcome
Gradients vary by many orders of magnitude
Shallow basins
with small slopes
Deep Valleys with steep slopes
Types of Optima
Types of Systems

Figure 3.14 Difficulties in Solving Nonlinear Optimization Problems (After Shang, 1997).

Figure 3.15 represents a multiple model approach to engineering optimization. By employing appropriate axes of demarcation, a nonlinear problem can be subdivided into smaller regions (regions 1, 2 and 3). Different optimization techniques can then be applied to solve each of the regions. The optimal solution is therefore developed from these 3 sub-solutions. In the mine design and optimization process, the subdivisions are done on the basis of the natural processes underlying the various phenomena to be modelled/designed and/or optimized. For example in the design of the pit, the ore body is modelled using geostatistics. In order to make reliable predictions from the model, machine learning (neuro-genetic) algorithm is used to build an intelligent block predictor.

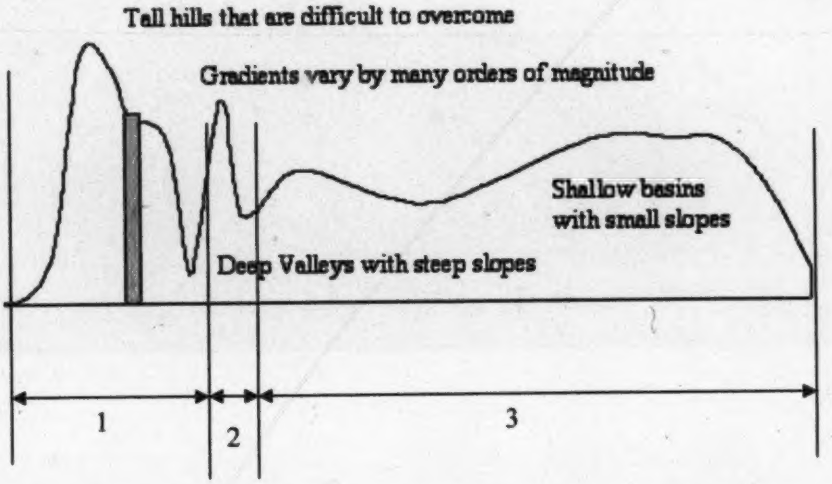

Tall hills that are difficult to overcome
Gradients vary by many orders of magnitude
Shallow basins with small slopes
Deep Valleys with steep slopes
1
2
3

Figure 3.15 Multiple Model Approach to Optimization of Nonlinear Systems

Optimization is the principle underlying the analysis of many nonlinear systems, complex systems, decision and resource allocation problems. Optimization can simply be defined as finding the maximum or minimum of a certain function or system defined on some domain. An optimization problem consists of a set of unknown variables, an objective function and a set of constraints with associated set of solutions. Some optimization problems are not constrained. Optimization theory can be considered as a body of mathematical and numerical algorithms, which is employed in identifying the best candidate from a number of alternatives. This best candidate (optimized solution) is chosen without having to evaluate all possible alternatives. In optimization methods, the best results are obtained by using mathematical, heuristic and iterative algorithms. In view of the intensity and complexity of the optimization processes, most optimization algorithms are computerized.

Classical theories of optimization are differential calculus, variational calculus and optimal control theory. Generally only a single criterion of optimality is employed. Some of the vast ranges of choice of optimality criterion are total capital cost, annual cost, return on investment, cost-benefit ratio, net present value, minimum production time and maximum pull down force[Anon, 1995].

The best candidate must however be the minimum or maximum value of the chosen performance index. The boundaries of the engineering system must be well defined to aid in the selection of the best model. A system which is described as the area of interest in Figure 3.16 is a restricted portion of the universe. The universe is the unconstrained solution space. The system boundary thus separates the area of interest from the universe. The quantitative criterion of optimization must also be defined. It is considered impossible to find a solution that maximizes profitability and minimizes cost simultaneously. A choice is normally made between profit maximization and cost

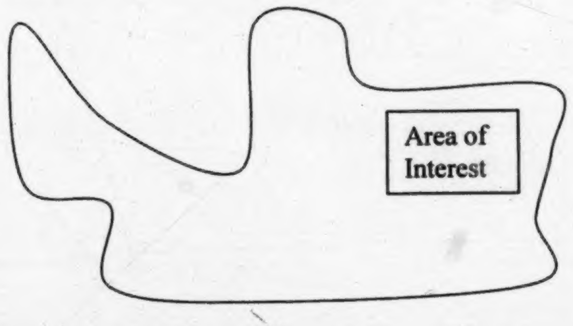


Figure 3.16 A System Within a Universe

minimization in optimization problem formulation. The model thus expresses the manner in which the variables are related. An optimization problem P is a 4-tuple $<I_P, S_P, f_P, Opt_P>$, such that for each pair of $x \in I_P$ and $y \in S_P(x)$, $f_P(x,y)$ is a real number and $Opt_P \in$ (max,min} defines the problem as maximum or minimum. For a given input instance $x \in I_P$, the optimization problem P is solved by finding the required solution y in $S_P(x)$ which optimizes the value of the objective function $f_P(x,y)$ among all other solutions in Opt_P. [Anon, 1995].

3.5.1 Combinatorial Optimization

In the framework of computational theory, combinatorial optimization problems are typically finite and the best results are chosen from a finite number of possibilities. It is however true that these possibilities may include all trees on n nodes and/or all Hamiltonian circuits of a complete graph. Exploring all the possibilities of finding the best solution for a small problem is practically impossible [Grötschel, 1993].

3.5.2 Global Optimization

Optimization can be local or global depending on the way that particular algorithm works (Figure 3.17). Nonlinear optimization problems generally contain a large number of local optima. These optima can be classified as weak or strong depending on their relative importance to the global optima. One of the strong optimum solutions invariably qualifies as the global optimum. Global search methods must be able to overcome local optima in order to obtain a global optimum. Solution methods for nonlinear optimization problems can be local and global.

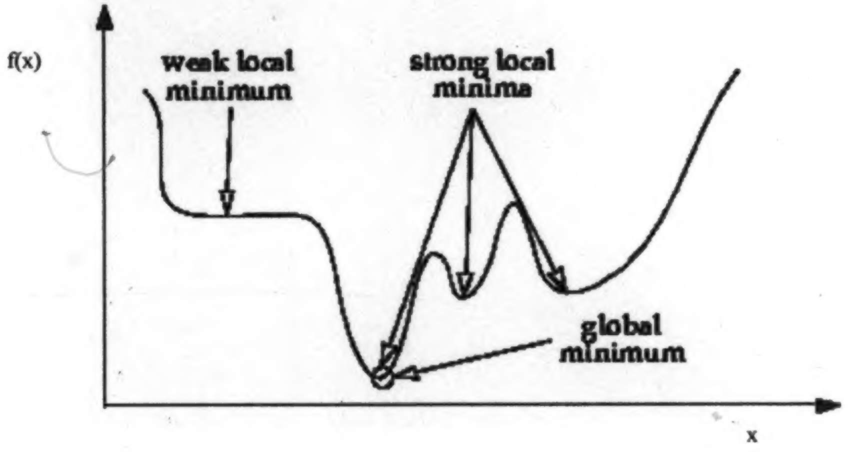


Figure 3.17 Local and Global Optima for Unconstrained Optimization Problems (Anon, 1995).

Local optimization methods (simplex, gradient-descent and Newton), use local information (gradient or Hessian) to achieve local minimum (Shang, 1997). Local optimization methods are best suited for uni-modal problems. However, global optimization methods employ heuristics, gradients and Hessians to get out of local optima into the global optima. Even though local optima may be good for some problems, many engineering applications require global optima. Many nonlinear problems have multimodal objective functions with many local optima. In an unconstrained optimization problem as depicted in Figure 3.17, all the optima are considered during the solution to the problem. Strong local minima is the set of all the well-defined minima. This includes the global minima. The global minima is the absolute minima. The global optimum solution is therefore the true optimum. The situation may be different in a constrained optimization problem. The global optimum is the optimum solution within the constraints (see Figure 3.18). in this case the solution space lies to the right of the constraint. Though the optimum point to the left of the constraints is the actual optimum if the constraint is removed, it is not considered at all (Anon, 1995).

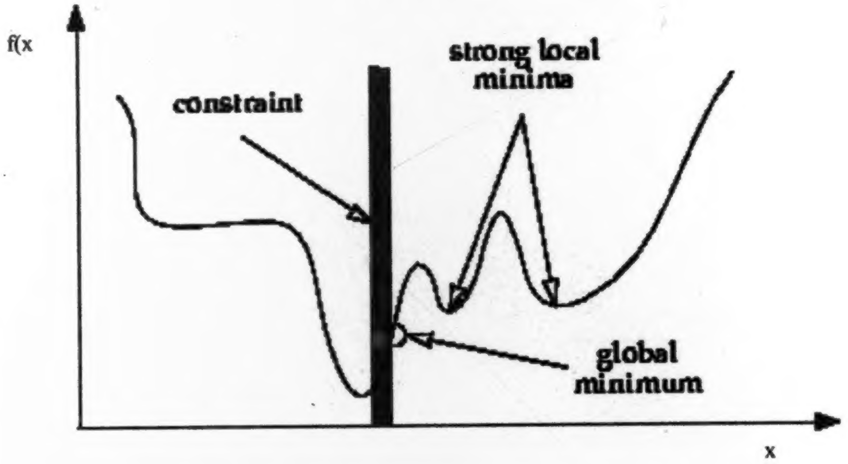


Figure 3.18 Local and Global Optima for Constrained Optimization Problems (Anon, 1995).

3.5.3 Stochastic (Probabilistic) Global Optimization

There are a large number of real world optimization problems which cannot be satisfactorily solved by discrete optimization search algorithms like gradient descent and Newton's methods. Thus these global optimization methods are weak in their local and global searches. The global minimum is this case is within the set of local minima. Random search or stochastic techniques therefore become the methods of choice (see Figure 3.19). Stochastic/probabilistic global optimization employ probability theory to solve nonlinear optimization problems. In the simplest form, stochastic search methods employ a series of restarts to get out of local into global optima. Stochastic optimzation methods are employed in solving large-scale problems. Within the context of the global search, the same strategies are employed in solving unconstrained and constrained problems. The pure random search methods are namely random line search, adaptive line search, adaptive random search, evolutionary algorithm, tabu search and simulated annealing [Anon, 1995 and Shang, 1997].

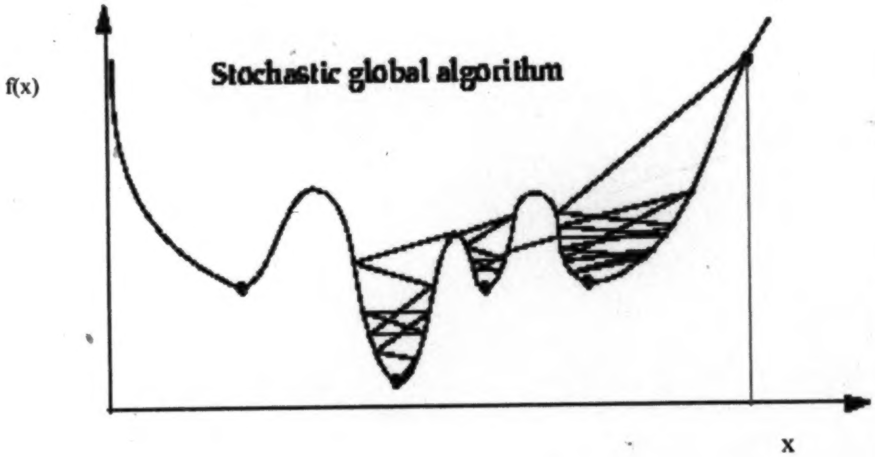


Figure 3.19 Stochastic Global Optimization (Anon, 1995).

3.6 Conclusions

In this chapter, conceptual frameworks which can be applied to the design and optimization of an open pit mine are discussed. Even though some of these frameworks are theoretical in nature they will be put to use in one form or the other in the following chapters. The importance of the correct design and optimization of an open pit mine to the success of future mining operations can not be overemphasized. Mine operators, financial institutions and other stakeholders, depend on the results of an open pit design and optimization to make crucial decisions. A general approach for solving the design and optimization problem is a definition of the problem followed by the design process and then the optimization algorithm. The output is checked against the problem definition to see whether it is satisfactory. The design problem is divided into slope design and block modelling, each of which requires a different solution methodology. Optimization involves the application of mathematical, numerical, heuristic and machine learning algorithms to identify the best candidate solution from a number of alternatives. Various optimization methodologies have been developed to solve a diverse number of problems. However most real world problems cannot be satisfactorily solved by discrete methods such as gradient descent but rather by stochastic methods such as simulated annealing, genetic algorithms, tabu search, random line search and adaptive line search. The particular frameworks that will be applied to the various aspects of the design and optimization will be discussed in the following chapters.

CHAPTER 4

MATHEMATICAL MODELLING USING MACHINE LEARNING

This thesis is aimed at designing and optimizing an open pit mine. This chapter therefore deals with the mathematical foundations of the various design and optimization algorithms employed. The mathematical models will cover slope design, geostatistical ore reserve estimation, simulated annealing, neural networks and genetic algorithms.

4.1 Open Pit Optimization Modeling Paradigm

The fundamental paradigm in open pit mine design and optimization is the maximization of net present value of the ultimate pit. The optimization algorithm must select the set of blocks which maximizes the net present value when mined together over the life of the mining operation. For a n-period pit, the maximum discounted net present value can be defined as;

$$Max.NPV = \sum_{t=1}^{n} \frac{(1-Tx_t)(\mathrm{Re}v_t - TC_t - R_t - NCC_t) - CC_t + NCC_t}{(1+i)^t} \qquad 4.1$$

Subject to FS > 1.25(slope constraints); Price > Cost (economic viability) and

$$\sum_{T=1}^{n}(Y_T + W_T) \le N_t$$

Due to the time value of money represented by a discounting factor, cash flow earned in the earlier years of the mining operation have a higher impact than that earned at later years. However the scheduling aspect of an open pit operation is beyond the scope of this work. Employing the number of ore and waste blocks mined from years 1 through n, the first part of equation 4.1 becomes;

$$Max.\ \ NPV = \sum_{t=1}^{n} \frac{(1-Tx_t)\left(P_t * \eta_{av} * \sum_{j=1}^{T} g_j * Y_j - \sum_{j=1}^{T} c_j * Y_j\right)}{(1+i)^t} + \frac{(1-Tx_t)\left(-\sum_{q=1}^{T} c_q * W_q - OC_t - R_t\right) - CC_t + NCC_t}{(1+i)^t} \qquad 4.2$$

Air blocks (blocks with values of zero) have been omitted since they have no effect on the net present value.

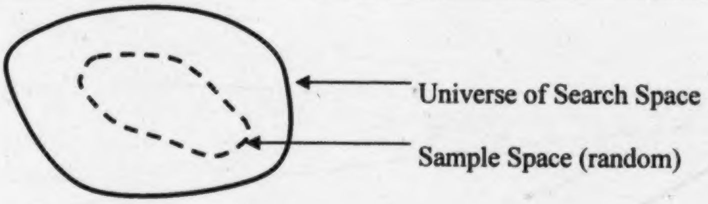


Figure 4.1 Relationship Between Sample and Universal Spaces.

4.1.1 Design Paradigms for Machine Learning Algorithms

It is assumed that a finite hypothesis space represented as the universe or search space (H) exists with a single truth hypothesis H(x). It is further assumed that the sample space is randomly selected and falls totally within the universe. The data points are representative of the search space and are unbiased. Machine learning involves choosing a hypothesis/learning algorithm, h(x) based on sample data such that $H(x) \equiv h(x)\ \forall\ x$ in X.

4.1.2 Hypothesis Testing

The accuracy of the chosen hypothesis can be easily evaluated if there is abundance of data. On the contrary, the absence of adequate data introduces bias and variability in the estimates from a hypothesis. Bias is defined as the difference between the value of the estimator ($err_s(h)$) and the true value/underlying population to be estimated ($err_d(h)$) and is given by the equation,

$$bias = E\left[\bar{err}_s(h)\right] - err_d(h) \qquad 4.3$$

If S is the training set used to produce h, then h and S must be chosen independently in order to obtain an unbiased estimate. The variance of the estimate is defined as the difference between the measured accuracy and the true accuracy for an unbiased estimator. The measured accuracy may still vary from the true accuracy even if random test samples are used to evaluate the hypothesis. The variance is known to be indirectly proportional to the sample size.

The true error of a hypothesis h with respect to a target function, H and a distribution d is the probability of h mis-classifying an instance drawn from d. Thus

$$err_d(h) \equiv \underset{x \in d}{\Pr ob}[H(x) \neq h(x)] \qquad 4.4$$

Also the sample error of h with respect to the target function, H and the sample data, s is the proportion of samples mis-classified by h. This is given by the equation;

$$\overline{err}_s(h) \equiv \sum_{x \in s} \partial[H(x) \neq h(x)] \qquad 4.5$$

where

$$\partial[H(x) \neq h(x)] = \begin{cases} 1 & if \quad H(x) \neq h(x) \\ 0 & otherwise \end{cases} \qquad 4.6$$

As depicted in Figure 4.2, if by careful analysis and modelling, the chosen hypothesis is equivalent to the true function, then the right hand side of equation 4.4 becomes zero.

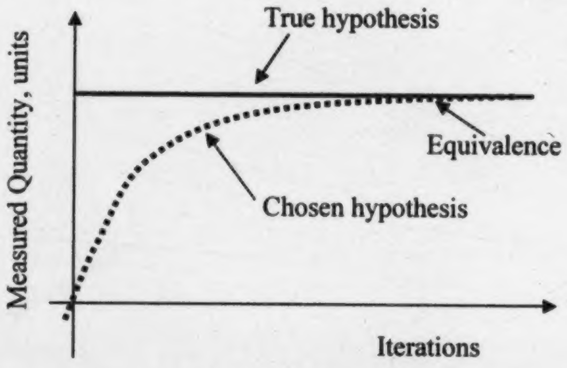


Figure 4.2. True and Chosen Hypotheses

In reality the function of the chosen hypothesis approaches the true hypothesis assymptotically. Suppose that the data samples are drawn independently of one another and of the discrete-valued hypothesis, h then the N% confidence interval of the true error is given by;

$$err_d(h) \approx \underset{s}{\bar{err}}(h) \pm z_N \sqrt{\frac{\underset{s}{\bar{err}}(h)\left(1-\underset{s}{\bar{err}}(h)\right)}{n}} \tag{4.7}$$

4.2 Mathematical Modelling by Adaptive Logic Networks (ALN)

Artificial neural networks have become one of the tools used by researchers to model complex systems, networks, and physical phenomena that do not lend themselves to accurate classical mathematical representation and analysis. According to Philip (1989), artificial neuron was initially designed to mimic the first order characterization of the biological neuron. Some researchers believe that problems that can not be modeled by partial differential equations, graph theory, heuristics and operations research techniques can be adequately modeled and solved using artificial neural networks and or fuzzy logic.

Neural networks have been successfully employed in pattern recognition or classification and function approximation. Like the open pit optimization problem, most of the problems that are now been solved using neural networks are described as computationally intractable.

4.2.1 Perceptron Network

The perceptron is a paradigm that requires supervised learning based on a set of examples of required or proper network behavior. Suppose we have a set of variables x_i and t_i, given as:

$(x_1,t_1), (x_2,t_2), (x_3,t_3), (x_4,t_4),\ldots(x_i,t_i),\ldots, (x_q,t_q)$ 4.8

where x_i is an input and t_i is the corresponding target/desired/true input. The error function is defined by;

$$e = t - y \quad 4.9$$

where t is the target output and y is the real output. The learning rule for the weight vector is given by the equation,

$$W_1^{new} = W_1^{old} + eX = W_1^{old} + (t - y)X \quad 4.10$$

Where W_1^{new} is the new weight vector; W_1^{old} is the old weight vector and X is the matrix of variables. This rule can be extended to train the bias by noting that a bias is simply a weight whose input is always 1 and its weight is 0.

The perceptron rule in matrix notation is written as

$$W^{new} = W^{old} + eX^T \quad 4.11$$

$$\theta^{new} = \theta^{old} + e \quad 4.12$$

Where e is the error vector and eX^T is the outer product of the matrix. The perceptron will always converge to a solution in finite number of steps if one exists for the problem under investigation. The perceptron learning rule of Frank Rosenblatt has been successfully used to classify input vectors that can be separated by a linear boundary. Such vectors are called linearly separable. In situations where the problem is not linearly separable, the single layer perceptron can not be used. In such situations a multi-layer perceptron together is used.

4.2.2 Adaptive Logic Network (ALN)

The adaptive logic network, which was invented in 1974 (Armstrong and Bochmann, 1974) and improved thereafter, is essentially a variant of the ADALINE. It is considered a feedforward neural network formed from perceptrons or linear threshold units which send their output to a tree of units that realize OR and AND logic gates. It is an effective tool for approximating any continuous real-valued function on N-dimensional space given a set of sample points.

An ALN can be considered a linear combination of a column vector of parameters and an input or predictor variable vector. As a feature of neural network, the initial component of the input vector is made equivalent to 1 in order to allow for a constant term or bias. Each of such hyperplane is referred to as a linear piece. Node values are calculated as either a maximum (logic OR) or minimum (logic AND) of at least one leaf node in the next layer of the tree. Subsequent layers are calculated in the same way until the final layer, which computes the models scalar output y. Any continuous function can be approximated to any desired accuracy by a piece-wise linear approximant - ALN with a suitable architecture (layers, nodes, node types) and coefficients (weight vectors in the leaf layer). The simple max-min (OR-AND) structure of an ALN makes it computationally efficient in modeling non-linear systems. Once the ALN has learnt its pattern recognition, computational or functional representation from the data set, it could be used to predict future behavior or control the functioning of the equipment in the required fashion.

ALNs represents complex functions by piecewise linear functions. The general expression for a linear function is

$$x_m = w_o + w_1x_1 + w_2x_2 + w_3x_3 + \dots\dots\dots\dots + w_{m-1}x_{m-1} \qquad 4.13$$

where x_m is the output variable, $x_1 \dots x_{m-1}$ are the input variables, w_i is the sequence of weights and m is the dimension of the problem. The expression representing an ALN is formed using a tree expression of maximum and minimum operations, corresponding to the OR and AND units in the ALN tree to form the piece-wise linear function: shown as a thick curve (see Figure 4.3). ALNs can be viewed from 2 different directions. First an ALN can be considered as a network that computes a logical value which indicates whether the following inequality is true;

$$x_m \le f\left(x_1, x_2, \dots\dots\dots \quad , x_{m-1}\right) \qquad 4.14$$

given m real-valued inputs including x_m. An ALN can also be considered as a neural network which computes a real-valued function;

$$x_m = f(x_1, \ldots\ x_{m-1}) \qquad 4.15$$

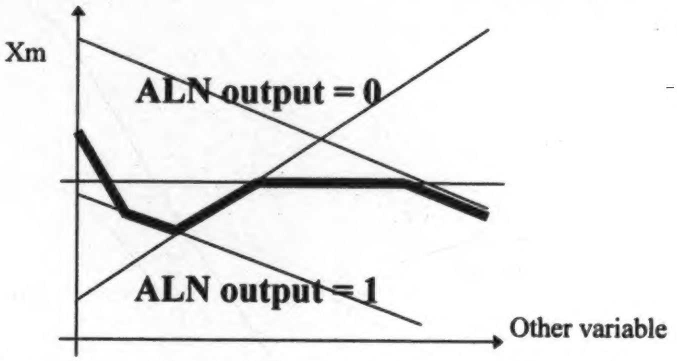


Figure 4.3 Representation of a Function by ALN (After Armstrong et al., 1995).

Least squares fitting/linear regression is used to fit the linear piece on the data point in a certain portion of the input space. In order to produce real values from the logical values, AND and OR gates are converted to minimum and maximum nodes respectively. Linear threshold units are also converted to linear functions. Linear functions of the form,

$$x_m = w_0 + \sum_{i=1}^{m-1} w_i x_i \qquad 4.16$$

According to Armstrong, over the years three distinct versions of the ALN concept have evolved, namely versions 1, 2 and 3 ALNs. The adaptive logic network, which was invented by Professor W.W. Armstrong (University of Alberta) is essentially a variant of the ADALINE. Once the ALN has learnt its pattern recognition, computational or functional representation from the data set, it could be used to predict future behavior or control the functioning of the equipment in the required fashion.

4.2.3 Network Architecture of ALN

ALNs differ from other multilayer perceptrons in that logic gates are used in all the hidden layers except the first. Version 1 ALNs are essentially limited to Boolean inputs, whereas version 2 ALNs involve networks of logic gates whose inputs are computed by fixed thresholds on the continuous input variables. Version 2 ALNs are adequate for sensor measurements, for instance. A measured value greater or less than a fixed threshold is input into the neural net as a 1 or 0 respectively. A version 2 ALN adapts any of the four kinds of functions - AND, OR, LEFT and RIGHT, at the nodes of the logic tree. Supynuk (1992) successfully used a version 2 ALN for control applications. In spite of their potential usefulness, version 2 ALNs are difficult to encode and decode, giving rise to difficulties in understanding them and therefore casting doubts on their usefulness. On the other hand, version 3 ALNs employ a logic network of ANDs and ORs with input from LTUs. A version 3 adaptive logic network is essentially a type of feed-forward multilayer perceptron which employs linear threshold units in a first hidden layer, and logic gates AND and OR in the other hidden layers in the output layer (Armstrong, et al, 1995). An example of version 3 ALN and an illustration of how the ALN represents a function are shown below.

According to Armstrong, the units of an ALN form a tree with a single boolean output. As shown in Figure 4.3 the output of ALNs is a zero or 1. A is one if the point is on or under the graph of a function, and a zero otherwise. Other characteristics peculiar to version 3 ALNs are summarized below;

(1) Partial derivatives or arbitrary bounds can be imposed on functions during the training stage. The learned function can be weakly or strongly monotonic increasing (or decreasing) in any variable.

(2) As mentioned earlier the open architecture of the logic tree can be used to impose qualitative constraints such as convexity on parts of learned functions. Qualitative apriori knowledge in the form of physical laws can be employed.

(3) A separate program evaluates the function trained by the ALN.

(4) There is no black box involved. A learned piecewise function can be checked for conformity to a specification by examining all the blocks of the input space determined by the decision tree and the simple combinations of linear functions/pieces using maximum and minimum operations that compute the output of that particular block.

(5) A different output variable can be used to convert the results of an ALN training into a DTREE computation. A function inverse is obtained in this way. The variable must be one in which the original ALN output was forced to be monotonic.

Larger ALNs can be used to represent a continuous function to any required precision uniformly on a compact set. In this case, Riemann sums are employed to approximate the area under the continuous function. The only modification required is to slant the edges of each pillar slightly so that a function is obtained which is represented by a two-layer ALN in the form of an OR or ANDs of linear functions. The weight on the output of each of the linear pieces can be normalized to -1. Though other neural networks have adopted curvi-linear functions in their computation, ALNs use linear functions for the following reasons;

- using linear pieces to fit training point can easily be conceived and a linear approximation resulting from such fitting is reliable and easy to accomplish;
- using synthesized functions require computing values that deviate a lot from the training points. A linear extrapolation is reasonable in such a situation;
- ALN Decision Tree can be used to evaluate the piecewise linear function at a high speed.
- The monotonicity of each linear piece is completely controllable. Bounds can even be placed on the rate of change of the output with respect to each input. The user can also have a great influence on the qualitative properties of functions produced by ALNs.
- On creating an ALN Decision Tree, one can obtain all the weights of all the linear pieces and information on how they are put together to form the learned continuous function. This enables one to analyze the results of learning for accuracy or conformity to a specification.

4.3 Mathematical Modelling by Continuous Adaptive Time Neural Networks

The continuous adaptive time neural network (CATNN) is analogous to the time delay neural network (TDNN), except that the look back intervals are adaptive in the CATNN. The look back in CATNN is interpolated smoothly across records (BioCOMP Systems, 2000). The output of neuron is given by;

$$O_i(n) = \begin{cases} x_i(n) & \text{if i belongs to A} \\ y_i(n) & \text{if i belongs to B} \end{cases} \qquad 4.17$$

For j Є B, the net internal activity of neuron j at time n is given by;

$$p_j(n) = \sum_{i \in A \cup B} w_{ji}(n) O_i(n) \qquad 4.18$$

The output of the neuron j, for the next (n+1) step is computed by running $p_j(n)$ through the non-linearity ψ(.),

$$y_j(n+1) = \psi(p_j(n)) \qquad 4.19$$

The jth element of the error vector is defined as,

$$e_j(n) = \begin{cases} d_j(n) - y_j(n) & \text{if } j \in \xi(\text{n}) \\ 0 & \text{otherwise} \end{cases} \qquad 4.20$$

The set of visible neurons as specified by ξ(n) can be varying over time. Thus the instantaneous value of the sum of squared error is at time n,

$$\xi(n) = \frac{1}{2} \sum_{j \in \xi} e_j^2(n) \qquad 4.21$$

The main objective is minimization of the total cost function which is defined as the value of the ξ(n) computed over all time;

$$\xi_{total} = \sum_n \xi(n) \qquad 4.22$$

Since we are using the method of steepest descent, the cost function is differentiated with respect to the weight vector w_{ji},

$$\frac{\partial \xi_{total}}{\partial w_{ji}} = \sum_{n} \frac{\partial \xi(n)}{\partial w_{ji}} \tag{4.23}$$

Using chain rule to express the derivative of cost function ξ_{total} with respect to the weight vector,

$$\frac{\partial \xi_{total}}{\partial w_{ji}(n)} = \sum_{n} \frac{\partial \xi_{total}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{4.24}$$

Where the time index n runs over $v_j(n)$ and not $\xi(n)$. The first term on the right hand side (RHS) of equation 4.24 is considered as the change in the cost function ξ_{total} produced by a change in the internal activation potential v_j of neuron j at time n. The equality of the equation holds only when the RHS is summed over n. Thus,

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} \neq \frac{\partial \xi_{total}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{4.25}$$

The weights are updated using the ideal gradient descent, the learning rate η and equation 4.24,

$$w_{ji}(n+1) = w_{ji}(n) - \eta \frac{\partial \xi_{total}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{4.26}$$

The derivative of the activation potential $v_j(n)$ with respect to the weight vector $w_{ji}(n)$ for any given neuron j is equivalent to the input vector applied to neuron j.

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = x_i(n) \qquad 4.27$$

The gradient for neuron j is given by;

$$\partial_j(n) = -\frac{\partial \xi_{total}}{\partial v_j(n)} \qquad 4.28$$

Equation 4.26 can be re-written as,

$$w_{ji}(n+1) = w_{ji}(n) - \eta \partial_j(n) x_i(n) \qquad 4.29$$

The expression for the local gradient for neuron j is dependent on whether the neuron is in the hidden or output layers. These local gradients are given as;

$$\partial_j(n) = \begin{cases} e(n)\phi'(v_j(n)) & \text{neuron j is the output neuron} \\ \phi'(v_j(n))\sum_{m \in A} \Delta_m^T(n) w_{mj} & \text{neuron j is the hidden neuron} \end{cases} \qquad 4.30$$

and

$\Delta_m^T(n) w_{mj}$ is the inner product of the vectors $\Delta_m(n)$ and w_{mj}

Both of the vectors have a dimension of (m+1).

$$\Delta_m(n) = [\partial_m(n), \partial_m(n+1), \ldots\ldots, \partial_m(n+M),]^T \qquad 4.31$$

4.4 Mathematical Modelling by Generalized Regression Neural Networks

The generalized regression neural network which is also called the Nadaraya-Watson neural network can accept both discrete and continuous valued inputs. In this nonparametric approach, binary least squares or logistic regression methodologies are employed. In the binary regression case, two class cases are defined by the response y_i = 1 or 0 depending on whether the ith training vector x_i emanates from class 1 or 2. To indirectly estimate the posterior probability, a family of functions R is defined by the form (Holmstrom, et. al., 1996);

$$R = \{r(\bullet, t) : t \in T\} \qquad 4.32$$

The dot in the function r means the inner product. A function r is then selected that minimizes the sum of squared errors criterion,

$$\frac{1}{n}\sum (y_i - r(x_i))^2 = \min_{r \in R}! \qquad 4.33$$

The error function (equation 4.33) is actually an estimate of the expected value

$$E\left[(Y - r(X))^2\right] \qquad 4.34$$

Another approach is using a logistic regression methodology to model the posterior probability directly as in,

$$P(J=1 \mid X=x)=\frac{\exp(r(x,t))}{1+\exp(r(x,t))} \qquad 4.35$$

Given that $X_1=x_1,\ldots\ldots\ldots\ldots, X_n=x_n$, the unknown parameter t can be estimated by maximizing the conditional likelihood of $y_1,\ldots,y_n$.

The least squares fitting criterion in equation 4.33 can be thought as the results of using the maximum likelihood principle to estimate a regression model with normally distributed errors. The logistic model uses a binomially distributed error and is therefore statistically superior. These two approaches are easily mapped to situations more than two classes by using a one-of-c coding. If the response y_i is the unit vector $[0,\ldots\ldots, 0, 1, 0\ldots\ldots\ldots\ldots, 0]^T \in R^c$, with I in the j_ith place, the least squared method yields,

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c}\left(y_i^{(j)}-r^{(j)}(x_i)\right)^2 = \min_{r \in R}! \qquad 4.36$$

The conditional log approach is maximized by using the fitting criterion,

$$\sum_{i=1}^{n}\sum_{j=1}^{c} y_i^{(j)} \log q_j(x_i) = \sum_{i=1}^{n} \log q_{ji}(x_i) = \min_{r \in R}! \qquad 4.37$$

4.5 Decision Analysis in Design and Optimization

Decision theory is the amalgamation of probability and utility theories. A decision involves irrevocable allocation of resources and must therefore be based on adequate information, analysis and ordering of preferences and alternatives. If the present state of a system is unique, completely specified, with a complete knowledge of the outcome of an action and an unambiguous, totally ordered utility function, then decisions are simple and easy. However if the current state of a system is not precisely known, the effect of an action is probabilistic and the utility function is unspecified, the optimal decision is unknown. Conceptually, an optimal action can be defined by,

$$a^*(s) = \arg\max_a \left[U\left(\mathrm{Re}\,sult\{a,s\}\right)\right] \qquad 4.38$$

It is assumed here that a rational agent will choose an action or a set of actions that maximizes his expected utility and is strictly Pareto superior (or at least Pareto superior). The hypothesis H_M is Pareto superior to the hypothesis H_N if the outcome $O_i(H_M) \geq O_i(H_N)$ for every state i, with a strict inequality for at least some single value of i. Hypothesis H_M is strictly Pareto superior to the hypothesis H_N if the outcome $O_i(H_M) > O_i(H_N)$ for all i (Kreps, 1990).

Though the expected utility cannot be calculated directly due to the lack of adequate information, an agent can be made to maximize some performance measure to achieve the highest performance score; summing or averaging over the possible environments in which the agent is placed. This performance measure could be the maximization of the net present value of the final pit. If U(M) and U(N) are real valued functions that operate on states M and N, then U(M) > U(N) if and only if the axioms of utility are fulfilled and M is preferred to N. However if the learning agent is indifferent between M and N, then U(M) = U(N). The maximum expected utility of a lottery (complex scenario with probabilistic outcome) is some sum of the expected utilities. The equations representing the above utility theories are;

$$U(M) = U(N) \Leftrightarrow M \succ N$$

$$U(M) = U(N) \Leftrightarrow M \approx N$$

$$U(p_1, X_1; \ldots p_{n-1}, X_{n-1}, p_n X_n) = \sum_{j=1}^{n} p_j U(X_j) \qquad 4.39$$

In the case of outcomes described by two or more attributes, the multi-attribute utility theorem is employed. If the probability distributions of 2 actions M and N on an attribute A, are $p_m(a)$ and $p_n(a)$, respectively, M stochastically dominates N on A if

$$\forall a \qquad \int_{-\infty}^{a} p_m(a')da' \geq \int_{-\infty}^{a} p_n(a')da' \qquad 4.40$$

In this case N can be discarded.

In a sequential decision problem like open pit optimization, the reward for an action as well as the system environment dictate what sequence of actions an intelligent agent can take. An agent is a combination of computer program and machine learning architecture. The goals of an agent, its behavior and the percept sequence from the environment can be very complex. Simplifying assumptions can be used to reduce the level of complexity.

Suppose that the present state of an agent R is the block position ($S_t = S_{x,y,z} = 5,1, 5$) in a rectangular slice of 3-D blocks MNOP, depicted in Figure 4.4.

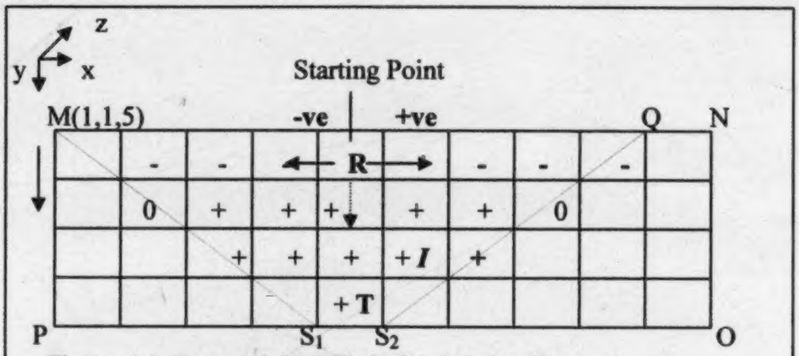


Figure 4.4. Sequential Motion of A 3-D Intelligent Agent Through A Typical Open Pit.

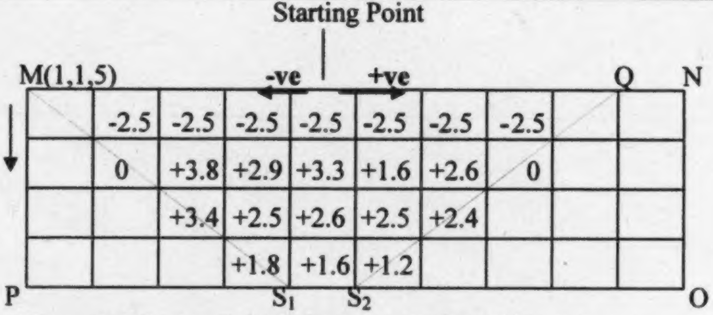
In view of the time value of money and the need to obtain a positive cash flow as early as possible, the agent will start at state $S_{x,y,z}$ (point R). The 3-dimensional agent can move to the right $S_{x+1,y,z}$ (positive) or left $S_{x-1,y,z}$ (negative), into $S_{x,y,z+1}$ (positive) or out $S_{x,y,z-1}$ (negative) of the paper and down $S_{x,y+1,z}$ (positive).

The surface topography is represented by MQN. The boundaries of the property are represented by MP and NO. The movements of the agent terminate on reaching T, the bottom of the pit. Lines MS_1 and QS_2 represent the slopes of the pit. The reward for an action (change of state) can be immediate, intermediate or cumulative. If the agent performs an action a, the probability of reaching the state j is given by the transition model equation;

$$T_{ij}^{a} = P\left(S_{t+1} = j \middle| S_t = i, A = a\right) \qquad 4.41$$

As in a standard planning problem, if the effects of an action and the values of the block are deterministic, then

$$T_{ij}^{a} \in \{0,1\} \qquad 4.42$$



	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5	-2.5		
	0	+3.8	+2.9	+3.3	+1.6	+2.6	0		
		+3.4	+2.5	+2.6	+2.5	+2.4			
			+1.8	+1.6	+1.2				

Figure 4.5 Sequential Motion of A 3-D Intelligent Agent Through A Typical Open Pit.

Generally the utility of an ore block is considered positive, that of a waste block is negative (dis-utility) and an air block (zero).

The immediate reward of an agent for an action is the value of the block it moves into. Using the block values in Figure 4.5, the rewards are given as;

$$R(S_{t+1}|S_t, A) = \begin{cases} -2.5 & if \quad S_{t+1} = [4,1,5] \\ -2.5 & if \quad S_{t+1} = [6,1,5] \\ -2.5 & if \quad S_{t+1} = [5,1,6] \\ -2.5 & if \quad S_{t+1} = [5,1,4] \\ +3.3 & if \quad S_{t+1} = [6,1,5] \end{cases} \qquad 4.43$$

The cumulative immediate reward becomes the utility function (U) of a sequence of actions and states. Assuming the utility of the block in front and that behind this slice is zero, the utility is given by,

$$\begin{aligned} U([s_0,a_1,s_1,a_1,.........a_n,s_n]) &= \sum_t R(s_{t+1}|s_t,a_t) \\ &= 14.70 \end{aligned} \qquad 4.44$$

Obviously the utility depends on the episode (ore, waste and air) and the maximum reward is obtained after a series of actions, not a single state. The best agent, given by the equation below is therefore the optimal multi-step agent which maximizes the expected utility by taking a sequence of actions.

$$\underset{[a_1,a_2,...,a_n]}{\arg\max} \sum_{s_0,...,s_n} P(s_0,s_1,.....,s_n|a_1,....,a_n)U([s_o,a_1,....,a_n,s_n]) \qquad 4.45$$

4.6 Simulated Annealing

Many real-world optimization problems like, 3-D open pit optimization, cannot be adequately solved by any of the above methods. Simulated annealing is a widely accepted stochastic iterative technique for solving a lot of combinatorial optimization problems. It is based on the principle governing the freezing of liquids or recrystalization of metals in the process of annealing. The melted material initially at a higher temperature cools to its lowest energy state at a temperature, T=0 via an adiabatic process. If the system temperature is too low or cooling is done very slowly, the cooling stops and the system is left in a meta-stable state (trapped in a local minimum).

The algorithm accepts changes that increase or decrease the objective function. A decrease in the function value is achieved through the use of probabilistic acceptance criteria. The probability of accepting decreases in the objective function values in a maximization process, decreases to zero according to the equation;

$$P = Boltzman\ \ Factor = \exp\left(-\frac{\delta u}{T}\right) \qquad 4.46$$

Where δu is an increase in the objective function, u and T is the control parameter. U can also be termed the cost function (Hamiltonian), which associates a cost with the sate of the system (temperature). The algorithm is able to avoid being trapped in local maxima.

Generally, there are 2 versions of the simulated annealing algorithm – homogeneous and inhomogeneous types, each of which represents a different way of decreasing the control parameter T (annealing schedule, cooling strategy or annealing scheme). In the case of the inhomogeneous algorithm, T is decreased after each transition and is therefore similar to a single inhomogeneous Markov chain. In the homogeneous case, T is decreased after a number of transitions L and is therefore similar to a sequence of homogeneous Markov chains each generated at a fixed value of T. The 5 major components of a simulated annealing algorithm are;

- the temperature function, T_n, where T is the temperature or control parameter and n is the number of times the parameter has changed,
- the repetition function L_n which is used to decide how many changes are to be attempted at each value of T,
- the probability density function, g(x) of the state-space of D parameters,

- the probability P of acceptance of the new net present value given the previous value,
- a stopping criteria which is to be used to terminate the algorithm.

4.6.1 Boltzmann Annealing

The probability of acceptance is based on the chances of obtaining a new state with net present value, NPV_{k+1} relative to the previous state with net present value NPV_k,

$$P(\delta NPV) = \frac{\exp\left(-{NPV_{k+1}}/{T}\right)}{\exp\left(-{NPV_{k+1}}/{T}\right) + \exp\left(-{NPV_{k}}/{T}\right)} = \exp\left(\frac{\delta NPV}{T}\right) \qquad 4.47$$

The above equation is referred to as the Gibbs distribution. A logarithmic temperature schedule can also be used, the temperature schedule is taken as,

$$T_K = T_0\left\{\frac{\ln k_0}{\ln k}\right\} \qquad 4.48$$

where k is the time index of annealing and k_0 is some starting index.

4.6.2 Simulated Quenching

An exponential schedule can be used to speed up the annealing time, namely;

$$T_k = T_0 \exp\{k(c-1)\} \qquad 4.45$$

where c is a constant and $0 < c < 1$.

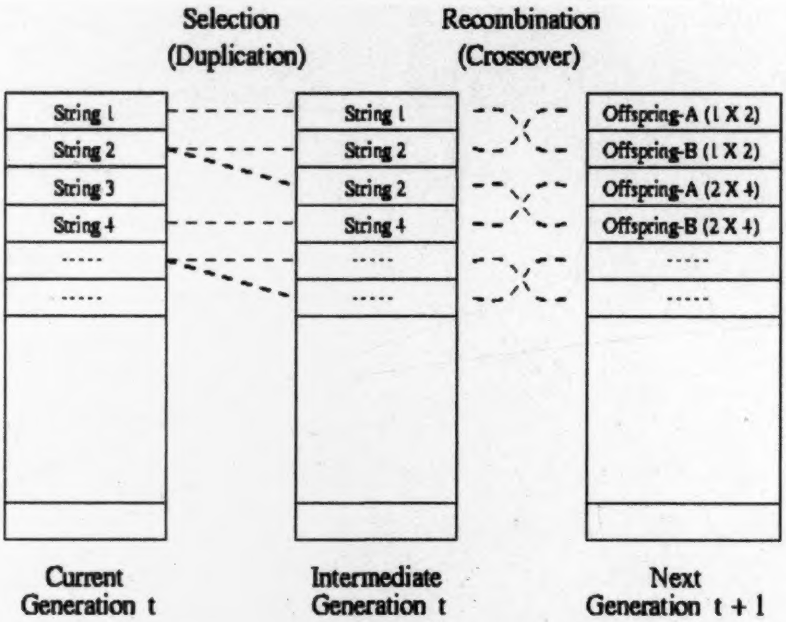
4.6.3 Fast Annealing

A Cauchy distribution can be used as the probability density function of the state space.

4.7 Genetic Algorithms

These class of algorithms which are also purely stochastic optimization algorithms employ the evolution theory of Darwin in computational modeling and simulation. They can be described as a stochastic directed search algorithm that can be employed in finding the global optima in static and dynamic environments. The problem solving methodology involves encoding the potential solution in chromosome-like data structure. Recombination operators are then applied to the resulting structures to preserve critical information (Whitley, 1995). The first step in the implementation of a genetic algorithm is the randomized generation of a population of chromosomes. The chromosome structures are then evaluated and allocated reproductive opportunities so as to enable the chromosomes which present a better to solution to survive.

Generally, a genetic algorithm can be described as any population-driven computational model that employs selection and recombination to produce new samples in an n-search space. The two major parts of a genetic algorithm are the encoding of the problem and the valuation of the function. Most applications of genetic algorithms involve non-linear functions. Its is however assumed that the variables representing the parameters of the function in question can be represented by bit strings. The variables are thus discretized to some power of two. The discretization must represent the underlying function. Since genetic algorithms can be classified as global search methods that do not employ gradient information in their search, non-differentiable functions and functions with multiple local optima are good candidates for this computational algorithm. In the context of the canonical genetic algorithm the initial developmental step is the generation of the first population of chromosomes by some random process. This is then followed by evaluation and the assignment of a fitness function to each string.


Selection
(Duplication)
Recombination
(Crossover)
String 1
String 2
String 3
String 4

String 1
String 2
String 2
String 4

Offspring-A (1 X 2)
Offspring-B (1 X 2)
Offspring-A (2 X 4)
Offspring-B (2 X 4)

Current Generation t
Intermediate Generation t
Next Generation t + 1

Figure 4.6: Process of Breakdown of a Generation into Selection and Recombination Phases and Assignment of Strings into Slots.

The evaluation is driven by the evaluation/objective function which provides the measure of performance which is transformed by the fitness into an allocation of reproductive opportunities (Whitley, 1993). In this case fitness is given by the equation;

$$fitness = \frac{f_i}{f_{av}} \qquad 4.50$$

where f_i is the evaluation associated with string i and f_{av} is the average evaluation of all the strings.

Commencing with the current population, an intermediate population is created through the application of selection (Figure 4.6). The next generation is then created by applying both recombination and mutation to the intermediate strategies generation.

4.8 Neuro-Genetic Algorithms

Even though neural networks have been known to be powerful tools for modeling nonlinearity, finding the best neural network has not been an easy task. For this reason, genetic algorithms have been combined with neural networks for automatic generation of the best network to solve a particular problem. These classes of hybrid algorithms are termed neuro-genetic algorithms.

4.9 Conclusions

In this chapter various mathematical models have been developed to address the algorithms employed in the design and optimization of an open pit mine. Most of these algorithms are machine learning algorithms. The prominent ones are the adaptive logic networks and the hybrid of neural networks and genetic algorithms. These mathematical representations will be reduced to computer models in the next chapter. In a situation where a commercial software exists which is based on the algorithm that software will be used in the place of coding.

CHAPTER 5

COMPUTER MODELLING

5.1 Introduction

The 3-D open pit optimization problem involves finding a set of air, waste and ore blocks which maximizes the net present value of a pit layout when mined together subject to slope and boundary constraints. The mathematical models described in chapter 4 are transformed into computer models in this chapter. Other models representing additional procedures, which are not part of the mathematical models in chapter 4, are also presented.

5.1.1 Transformation of Mathematical Models into Computer Models

As depicted in Figure 5.1, the three major design and optimization computer models are

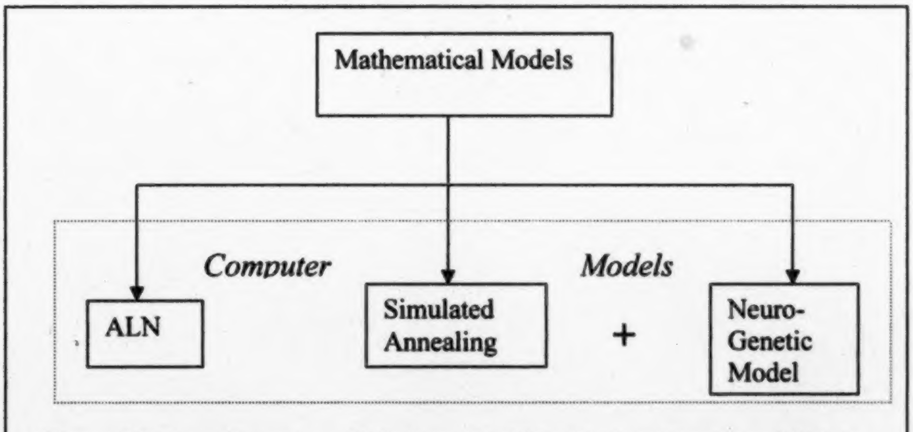

Mathematical Models
Computer
Models
ALN
Simulated Annealing
+
Neuro-Genetic Model

Figure 5.1 Transformation of Mathematical Models into Computer Models

adaptive logic networks, simulated annealing and neuro-genetic models. The various mathematical models in Chapter 4, representing adaptive logic networks, simulated annealing and neuro-genetics are transformed into computer models. The adaptive logic network models are on a stand-alone basis. However the simulated annealing model is complimented by the neurogenetic model in the pit optimization process. Geostatistical methods are employed in the block modeling and reserve evaluation.

5.2 Adaptive Logic Network Computational Algorithm

The computational algorithm for converting the sequence of equations into code and

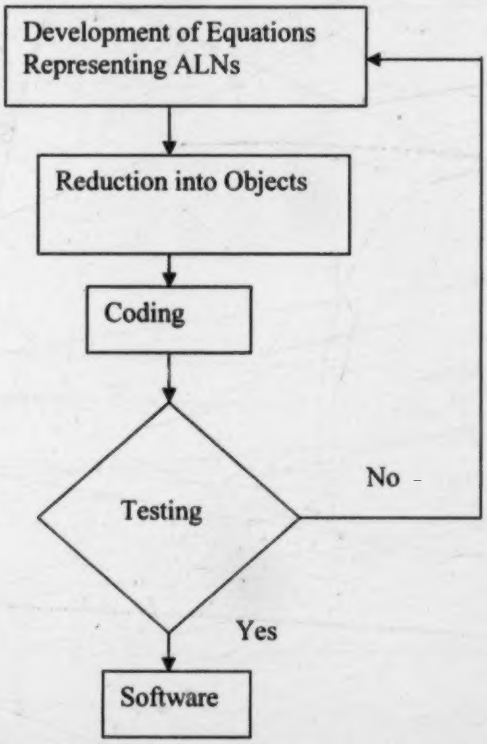

Development of Equations Representing ALNs
Reduction into Objects
Coding
Testing
No
Yes
Software

Figure 5.2 Flowchart for Transforming ALN Equations into Code/Software

software for the adaptive logic network is summarized in Figure 5.2. Initially, an arbitrary system of equations is developed to represent the ALN model. The equations and the processes involved in the modelling are reduced into various objects and then coded. The code is tested to see if it is accurate before being released.

The equations in the computer model representing the ALN are developed through the following steps;

Step 1: the development of an arbitrary number of linear functions of the form;

$$F_k = \sum_{j=0}^{n} w_{jk} X_j - Y \qquad 5.1$$

Generally some flexibility is built into the model through the introduction of a constant term in each function (i.e. $X_0 \equiv 1$). This is equivalent to the inclusion of a bias term in the neural network.

Step 2: assign values to weights (w_{jk}) and initialize it together with a threshold to a small random number.

Step 3: change the weights to produce the desired output.

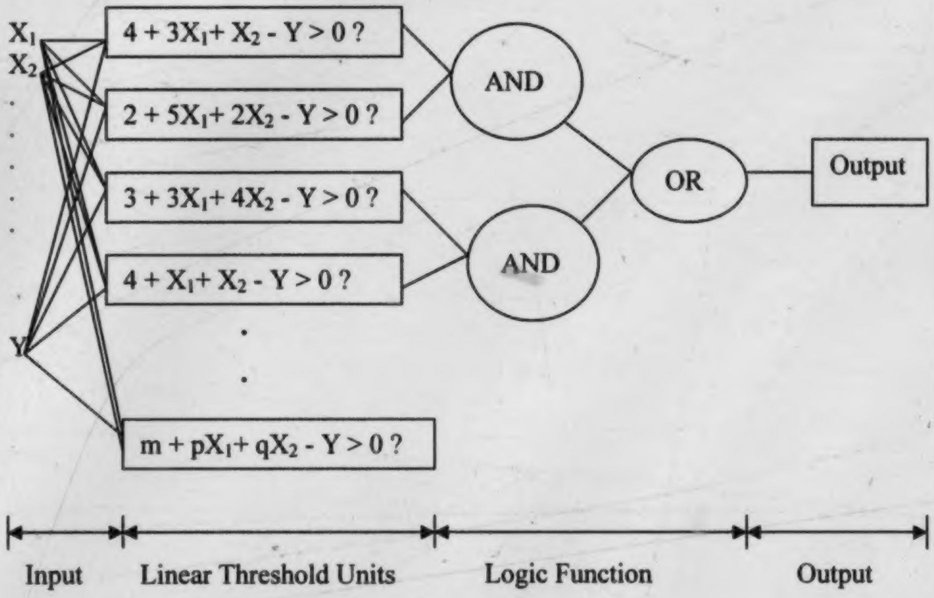
Step 4: check to verify whether $F_k = 0$. If $F_k = 0$ and n = 1, the model is defined by a straight line. If $F_k = 0$ and n = 2, the model is a plane. However if $F_k = 0$ and n > 2, the model is a hyperplane. Thus at a threshold of zero, the system of equations becomes;

$$F_k = \sum_{j=1}^{n} w_{jk} X_j \qquad 5.2$$

Step 5: the linear functions are then incorporated into the leaf node of a decision tree. Each leaf node evaluates whether $F_k \geq 0$, resulting in a true(1) or false(0) value. If the function evaluates to true, a threshold has not been reached and thus,

$$F_k \leq \sum_{j=1}^{n} w_{jk} X_j \qquad 5.3$$

for that particular combination of values for the input and output variables. The inclusion of linear threshold units in an ALN is central to its success. The logic term in ALN is the decision tree of AND and OR logic functions. Figure 5.3 describes the architecture of a three-layer ALN.



X_1
X_2
Y
$4 + 3X_1 + X_2 - Y > 0$?
$2 + 5X_1 + 2X_2 - Y > 0$?
$3 + 3X_1 + 4X_2 - Y > 0$?
$4 + X_1 + X_2 - Y > 0$?
$m + pX_1 + qX_2 - Y > 0$?
AND
AND
OR
Output
Input
Linear Threshold Units
Logic Function
Output

Figure 5.3 Computational Architecture of the Adaptive Logic Network

In this case only the LTU and logic functions are counted as layers. If the input and output layers are counted as well, this particular ALN can be described as a 5-layer neural network. The completed upper section of the ALN can be described by the following OR (AND(2), AND(2)) ALN structure. Even though the possible number of ALNs is infinite, only a small number of ALNs need to be considered. For n independent variables, each linear piece has (n+1) variable weights. The weights are also limited to the number of observations in the estimation sample. ALN employs sequential least squares as the estimation criteria during learning. The initial step in the ALN learning process is the random selection of parameter values. These values are then assigned to all the linear functions in the ALN. Apriori knowledge can also be used to assign parameter values. The next step involves the sequential input of training (input) values (X_t, Y_t) into the ALN. The resulting truth values are then propagated through the network to produce the tree output. After the first set of values has passed through the network, the next set of training pair is presented to it. A training epoch is completed after the entire data set has passed through the network. The rules of logic are employed in assigning formal responsibility to every logic node in an ALN. This enables the ALN to determine which linear piece is responsible for the output error. The output node is assigned the initial responsibility. The responsibility is then worked backwards. In this case responsibility is assigned to the nodes which will affect the value of the network's output node if they undergo a change in their truth values.

The software product so developed by Dendronic Decisions using object-oriented programming in C/C++ in a Microsoft Windows visual studio development environment.

Libaln is a library of object-oriented (C/C++) programs. The various adaptive logic network objects are listed in Appendix 5.1. Computer modelling in this case will be limited to the actual computations within this software environment.

Even though structured programming is employed in most engineering computations, there are number of other reasons why object-oriented programming is used in ALN software.

5.3 Computerization of Three-Dimensional (3-D) Optimization Algorithm

The optimization of a 3-D pit involves mining a sequence of blocks the total value of which maximizes the net present value over the life of the pit. The incremental policy iteration algorithm is employed. Let the current state of the open pit be the root of the tree of possible actions. This current state is defined by a set of blocks that form the minimum mineable 3-D pit (complete outline). These sets of blocks (ore, waste and air) form a 3-D ellipsoid of extraction (Figure 5.4). The geometrical model of the pit shell is constrained

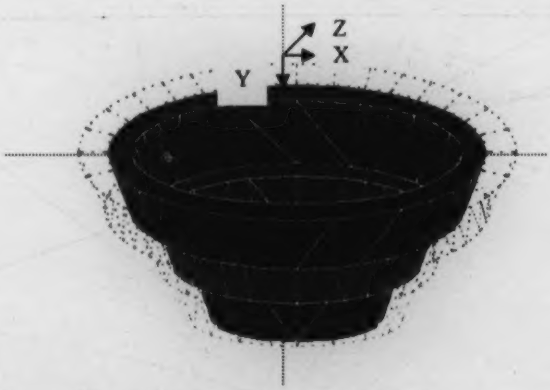


Figure 5.4 An Incremental 3-D Open Pit

by the slope angles of the pit walls. The pit shells can expand in all directions (Frimpong, et al, 1997). As depicted in Figure 5.5 in the present state there are 3 possible actions – movements of the intelligent agent.

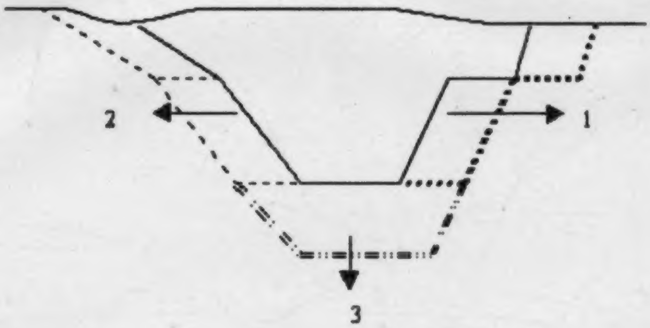


Figure 5.5 A Section Showing the Incremental Movement of the 3-D Intelligent Agent

Each action will result in one of three possible outcomes namely;

$$\delta NPV = \begin{cases} > 0\ \forall \text{ mining "ore" blocks} \\ = 0\ \forall \text{ mining "air" blocks} \\ < 0\ \forall \text{ mining "waste" blocks} \end{cases} \qquad 5.4$$

In the case of an increasing NPV (δNPV > 0), the intelligent agent mines a combination of ore or ore and waste blocks with a positive net benefit. When an equally-weighted ore and waste blocks are mined, there is no change in the net present value and the intelligent agent is considered to be mining air blocks.

In a situation where waste or a higher percentage of waste blocks are mined, the net present value is decreased (increased cost). The state of the net present value (NPV) is dependent on the set of blocks enclosed in the incremental pit outline.

```
function incremental policy iteration(R, δNPV) returns a policy
        inputs: R = transition model
                δNPV = incremental NPV
    local variables: NPV= net present value, initially equivalent to δNPV
                     P = incremental pit, initially optimal w.r.t NPV
        Repeat
                NPV ←—— Determination of Value (P, NPV, R, δNPV)
                Unchanged? ←—— True
                For each state i do
                        if $\max_c \sum_j R_{ij}^{c} NPV[j] > \sum_j R_{ij}^{P[i]} NPV[j]$ then
                $P[i] \leftarrow \arg\max_c \sum_j R_{ij}^{c} NPV[j]$
                unchanged?                False
                end
        until unchanged or end of blocks?
        return MAX PIT
```

Figure 5.6 Incremental Optimization Algorithm.

The outcome O(S, P) denotes the history/incremental tree starting from the state S and taking action according to the policy P. This can be considered as a random variable which depends on a transition model R. Thus the NPV of the state i is dependent of the historical state at the beginning of the action and the results of following the optimal policy;

$$NPV_{t+1}(i) = \delta NPV(i) + \sum_j R_{ij}^{Policy(i)} NPV_t(j) \qquad 5.5$$

In this case the algorithm works by picking a policy (incremental pit), then calculating the incremental NPV of this state using the set of block values in the state. It then updates the policy using the NPV of the successor states. This algorithm is represented in Figure 5.6.

5.4 Search Procedure and Situation Calculus of the Intelligent Agent

The issues that must be addressed by a typical search procedure are completeness, time complexity, space complexity, and optimality. Completeness addresses the issue of whether a particular strategy will find a solution provided there is one. Time complexity deals with the time it takes to find a solution. Space complexity addresses the memory requirements of the search. Optimality is concerned with whether the strategy employed will find the highest quality solution from a set of possible solutions.

In this case the entire block model is perceived as situations each of which is a snapshot of a state of the world. Present situations are generated from previous situations by actions of the agent. The situation argument is written as;

$$At\left(Agent\ ,[1,0,0]\right), S_0 \wedge At\left(Agent\ ,[0,1,0]\right), S_1\ ) \qquad 5.6$$

The results of performing an action is given by;

Result (Right, S_0) = S_1

Result (Down, S_1) = S_2

5.5 Tracking Location

The agent must remember where it has already traversed and what it saw there in order to deduce the location of ore, waste or air blocks to ensure that a complete evaluation of all the blocks occur. Though the initial location can be described by At(Agent,[1,0,0], S_0), it must know the angle in degrees (0 = x-axis, 90^0 in z-axis, and y in levels). Figure 5.7 depicts three movements of the agent.

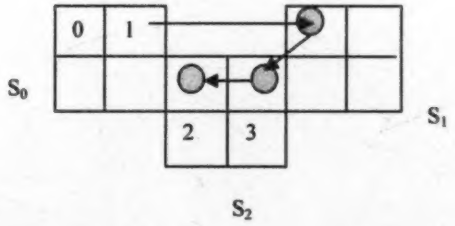


Figure 5.7 Location Tracking in An Intelligent Agent

5.6 Stochastic Optimization Methods

In the iterative deepening and widening algorithm, the search can be complicated if a large number of sections and/or subsections are involved. The search can also end up in a local maximum. Hence the need for other search methods. Consider the generalized global optimization problem outlined in Figure 5.8. A maximization problem involves locating the global optimum. Most of the discrete optimization algorithms like the iterative search; hill climbing and others may end up in the local maximum. Some of the stochastic optimization algorithms, which have been used to alleviate these problems, are random search, heuristic repair, random incremental improvement, Monte Carlo simulation and simulated annealing.

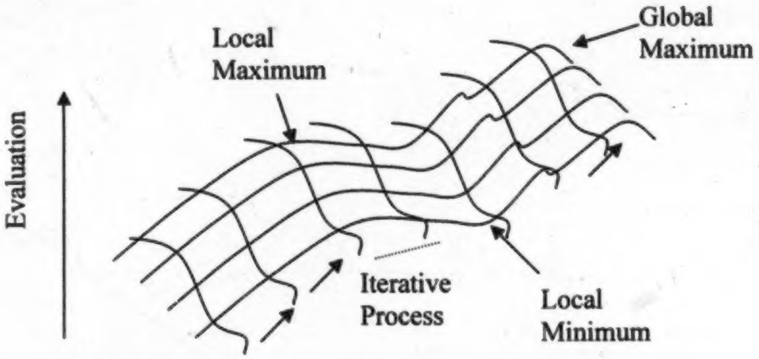


Figure 5.8 Dynamics of Iterative Improvement Algorithms

5.6.1 Random Search Algorithms

These algorithms use a prescribed probability density function or a set of probability density functions to generate a sequence of points in the feasible solution space. The basic random search algorithm generates independent and identically distributed random variables (points) from a single probability density. The random search algorithms are namely pure random search, random search, pure adaptive search and adaptive search.

5.6.2 Heuristic Repair

This iterative improvement algorithm involves assigning arbitrary values to all the variables. Modification operators are then employed to move them to the solution. A typical example of this class of algorithms is the min-conflicts heuristics, which selects values that result in the minimum number of conflicts with other variables (Russell and Norvig, 1995). It is very similar to the GSAT or Random Incremental Improvement algorithm presented below.

5.6.3 Random Incremental Improvement Algorithm

This is a random-restart, hill-climbing search algorithm. It starts by randomly generating a solution (Stephens, 1996). Random changes are then made to the solution. Each change to the solution is tested to see if there is improvement. The new solution is accepted if there is improvement. A stopping criterion should be imposed on the algorithm. One can stop the changes after a certain number of random changes. Generally for N items making N random changes may be sufficient. Alternatively, one can make changes until several test changes do not result in any improvement.

5.6.4 Monte Carlo Simulation

This type of numerical methodology uses a sequence of random numbers to perform statistical simulation. A probability density function is employed to describe the evolution of the physical or mathematical system. The simulation is achieved by randomly sampling the PDF and averaging over the number of observations. Generally the statistical error of the average can also be estimated.

The main components of a typical Monte Carlo algorithm are;

- A set of PDF's to represent the physical or mathematical system,
- A uniformly distributed set of random numbers on the unit interval,
- Some sampling rule utilizing the set of random numbers and the pdf's,

- The outcome of the quantities of interest must be tallied,
- Determination of the statistical error (variance) as a function of the number of trials and other quantities,
- The use of a variance reduction technique (for reduction of the variance in the estimated solution) to reduce the computational time of the Monte Carlo Simulation,
- the use of parallelization or vectorization algorithms/methods to enable efficient implementation on a computer.

5.7 Logic Flow Network for Simulated Annealing Algorithm

The mathematical formulae for simulated annealing will be transformed into algorithms and pseudo-code. Its will then be coded in MS Visual C/C++. The neuro-genetic algorithm is also developed from the combination of genetic and neural network mathematical models. BioComp's neurogenetic optimizer will be employed in the computer modelling. The output from the simulated annealing algorithm will be fed into the neurogenetic optimizer which will yield an actual optimized pit. In the simulated annealing paradigm, once an objective function has been established, the initial realization is modified so as to increase the value of the objective function in the maximization. The realization so obtained must be acceptably close to the target statistic. A flow diagram for the algorithm is depicted by Figure 5.9.

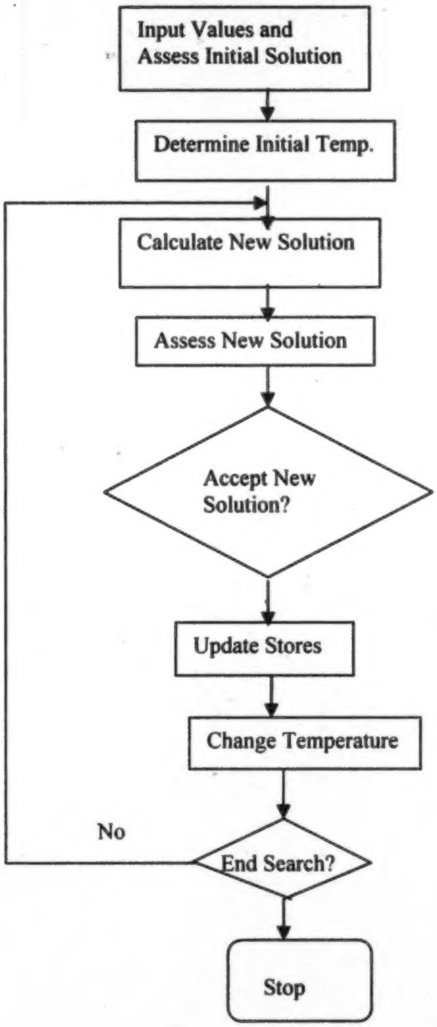

Input Values and Assess Initial Solution
Determine Initial Temp.
Calculate New Solution
Assess New Solution
Accept New Solution?
Update Stores
Change Temperature
No
End Search?
Stop

Fig 5.9 Flow Diagram of a Simulated Annealing Algorithm

```
Start
        Initialize (n, Tn, I, Ln) while n = 0;
        Repeat
            Repeat
                Generate state j a neighbour to i;
                Calculate $\delta NPV = NPV_i - NPV_j$;
                If Accept ($\delta NPV$, $L_n$) = = true then i = j
            until $L_n$ ;
            n = n + 1;
            Update $L_n$ ;
            Update $T_n$ ;
        Until StoppingCriterion = = true
    End

    Subroutine Accept($\delta NPV$, $T_n$)
        if $\delta NPV \leq 0$ then true
        else
            if random(0,1)<p(($\delta NPV$) then
                return true
            else
                return false
            endif
        endif
    End
```

Figure 5.10 Pseudo-Code for Homogeneous Simulated Annealing

5.8 Pseudo–Code Development for Simulated Annealing

The pseudo-code for a homogeneous simulated annealing algorithm is given in Figure 5.10.

5.9 Simplified Computer Program Development Process

The development steps of a typical computer program are definition of the problem,

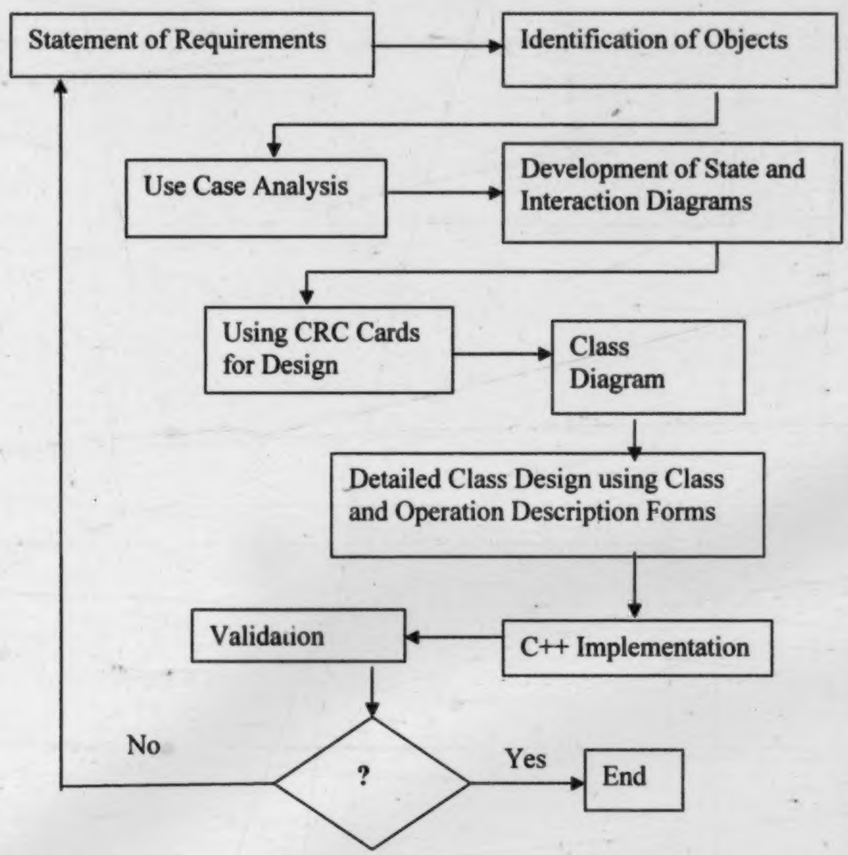

Statement of Requirements
Identification of Objects
Use Case Analysis
Development of State and Interaction Diagrams
Using CRC Cards for Design
Class Diagram
Detailed Class Design using Class and Operation Description Forms
Validation
C++ Implementation
No
?
Yes
End

Fig 5.11 Flow Diagram of the Object-Oriented Design Method

development of a solution, development of an algorithm, drawing of flowchart, creation of a pseudo-code, program coding, program debugging and documentation. In OOP, the process is a little different.

5.10 Object-Oriented Design of the Optimizer

The design and programming of the 3-D optimizer begins with a statement of requirements. This is followed by the identification of objects, use-case analysis and the development of state and interaction diagrams. The actual design process starts with the preparation of component-responsibility-collaborator (CRC) cards, the development of a class diagram and detailed class design using class and operation description forms. The resulting design is then coded in C++ language by employing Microsoft's Visual C++ 5.0. The entire process is represented by Figure 5.11. Characterization of software in terms of actions to be performed (behaviour) is central to object-oriented programming (Budd, 1997). In the initial stages, the behaviour of the entire software is characterized at a high level of abstraction. The behaviour of the various software subsystems is then described. Coding will only commence after the identification and description of all the behaviour.

5.11 Requirements Analysis

The software that is being designed will search through an array of 3-D block (block model) and pick a collection of ore, waste and air blocks to optimize the net present value of the pit when mined. The optimization process involves the use of stochastic simulation – simulated annealing as the search algorithm. The simulated annealing algorithm, its logic flow net as well as the pseudo-code is described in previous sections. A simulated annealing algorithm – SimAnnealOpt, has been designed and coded. The newly selected blocks will be added to those already in the bunch of selected blocks, to calculate the net present value according to equations 5.4 and 5.5. The computer will communicate the net present value of the ultimate pit to the user.

5.12 Identification of Objects for Open Pit Optimization Problem

The identification of objects and classes from the written problem specification is an important step in the OOP design process. An object type contains methods of procedures and functions together with data. Since objects are at the core of a successful OOP system, a clear understanding of which objects are to be included in the model is an imperative component of OOP. A program can be considered as a pool of objects.

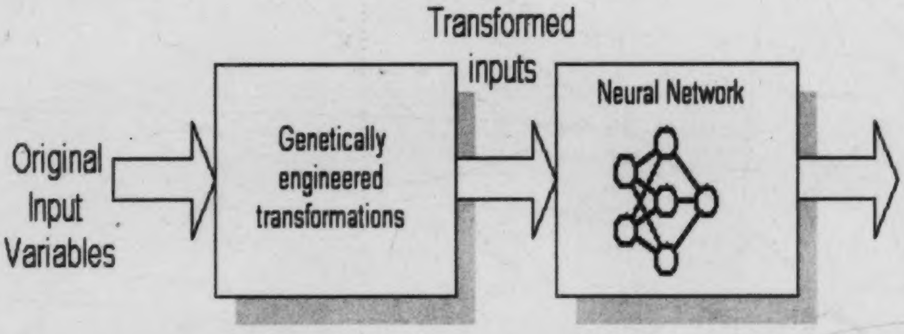


Figure 5.12 Using Genetic Algorithms to Configure Network Problems (After NeuroGENESYS, 2001)

Discovering objects is challenge faced by OOP designers and analysts. One way of discovering objects is the use of nouns and verbs in the project description as the candidate object. One should try to discover as many objects as possible and refine them the candidate object list later. The problem is examined in more detail once objects are discovered.

5.13 Neuro-Genetic Algorithm.

The use of genetic algorithms in combination with neural networks have been achieved via 3 processes, namely using the genetic algorithm to configure the problem for the neural network, using the genetic algorithm to configure the neural network architecture and using the genetic algorithm to train the neural network.

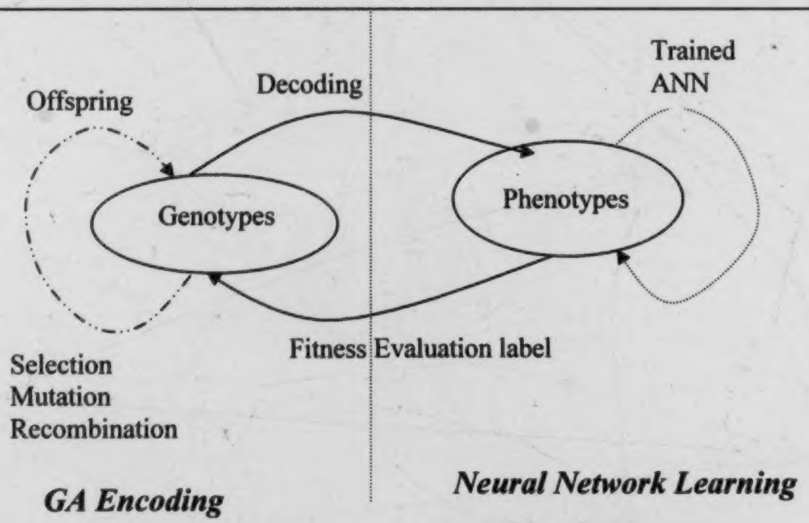

Trained
ANN
Decoding
Offspring
Genotypes
Phenotypes
Fitness Evaluation label
Selection
Mutation
Recombination
GA Encoding
Neural Network Learning

Figure 5.13 Using Genetic Algorithm to Design Neural Network Architecture

5.13.1 Using Genetic Algorithms to Design Neural Network Architectures

In this case the genetic algorithm is employed to transform the inputs or outputs of the problem. The transformed problem may be easier to learn, may require fewer weights or inputs or may aid in obtaining a better output. Such a process is outlined in Figure 5.12. This involves the use of a genetic algorithm to specify and optimize the network structure. This optimization addresses such network architectural issues as the overall topology, size and connectivity of the particular network.

Using genetic algorithms to configure and optimize neural networks involves the development of a population of candidate neural networks. Each individual neural network in the population is evaluated using the neural network criteria developed in sections 4.3 and 4.4 as well as the fitness function. This strategy enables the genetic algorithm to search for a single optimized neural network architecture for the particular application. A more elaborate explanation of the process is outlined in Figure 5.13. In this system a population of genotypes (an arrangement or a string of genes - genetic algorithm encoding scheme) is developed. Each gene is accorded values from a defined set of values. Genotypes are considered encodings of phenotypes or neural networks. The genetic code or genes represent numeric values or complex symbolic structures, which are translated into artificial neural networks by an appropriate decoding process. The resultant artificial neural network is tested using the available data and suitably defined performance measures or user-defined constraints. A fitness label is assigned to the artificial neural network after the evaluation is completed. Artificial neural networks are selected based on the fitness. Variety is introduced in the population of candidate artificial neural networks through the use of mutation and inversion. By repeating this processes over many generations, the population of neural networks evolves towards the king of the hill neural networks with highest fitness. In most cases the competing networks are also indicated with their performance scores. In the case of BioComps, neuro-genetic optimizer, other architectures are ranked behind the optimized neural network.

Thus a genetic algorithm can also be used to train the network. This actually involves find the best set of weights to optimize the network.

5.14 Computer Models for Computer-Aided Design (CAD)

The block models are developed using GSLIB software. The computerized mine design software employed are Gemcom and/or AutoCAD. The procedures used will be described in the experimentation process in Chapter 6.

5.15 Conclusion

In this particular chapter, the mathematical equations have been transformed into computer models. The major models are adaptive logic networks, simulated annealing and the neurogenetic optimizer.

CHAPTER 6

EXPERIMENTAL DESIGN AND EXPERIMENTATION

6.1 Introduction

This chapter is focused on experimental design and experimentation. In situations where appropriate software is available, it will be employed in the experimentation. Thus the models presented in Chapter 4 and the computer algorithms and programs described in Chapter 5 are employed in the design and optimization of an actual gold mine. The design of the various experiments, experimental setup and the experimentation are performed in this chapter. The gold data came from the Star Gold Mine, Zimbabwe (Frimpong, 1988) and the coal data from a Western Coal Mine in North America.

6.1.1 Machine Learning Algorithms

The various machine learning algorithms employed in this chapter are the back propagation neural network, continuous adaptive time neural network, generalized regression neural network, neurogenetic optimizer and the simulated annealing – neurogenetic optimizer. The backproagation algorithm (BP) employs gradient descent to train multilayer feed forward neural networks. An error function is defined in the final output of the network and weights are adjusted in the neural network to minimize the sum-of-squared error in the backpropagation algorithm. The generalized regression neural network (GRNN) stores the input and output variables in the network itself. It then looks at the difference between the current record and all the stored record to obtain an estimated output through interpolation. The continuous adaptive time neural network (CATNN) is a variant of time delay neural networks except that the look back intervals are changed by the connections as learning progresses. This makes it a little more advanced than time delay neural network. Time delay networks are somewhat similar to backpropagation neural networks. However there are multiple connections between input, hidden and output neurons. Each of these connections looks back to its weight to minimize the mean square error of the overall neural network. The neurogenetic optimizer combines the learning abilities of neural networks with the search properties of genetic algorithms to search through the data to uncover variables that contribute to the best models. It simultaneously searches through multiple neural network architectures and types to maximize the accuracy of the resulting neural network (BioComp, 2000). The simulated annealing – neurogenetic optimizer is a combination of the simulated annealing algorithm and the neurogenetic algorithm. Output from the simulated annealing algorithm is input into the neurogenetic optimizer to complete the optimization process.

6.2 Verification and Validation

Whereas verification deals with whether the design/system has been built right so as to meet the set of requirements or specifications, validation is concerned with determining whether the engineering process has produced the required product or design/system. Verification is the comparison of configuration items, components, subsystems, systems to their requirements to ensure correctness. There are several types of validity, namely operational validity, conceptual validity, requirements validity and design validity (Buede, 2000). Whereas conceptual, requirements and design validity are major players in the early part of the design, operational validity is employed in the latter stages. Conceptual validity exists if there is agreement between the system needs and the operational concept. Generally it is established at the initial stages of a design.

6.3 Experimental Design

In scientific studies, experimental design is meant to either depict the effect of a particular variable on the dependent variables or to extract the optimum amount of unbiased information from a process (Pukelsheim, 1993). In many scientific (practical and theoretical) problems, the relationship of interest is given as;

$$y = f(x,\theta) \tag{6.1}$$

Generally the experimental condition x is selected from the given experimental domain by the experimenter while the parameter system θ can be assumed to lie in the parameter domain. In order words, θ is controlled by nature whereas x is chosen by the experimenter. The type of function f, employed is instrumental to the success of the modelling process.

However due to the complexity of engineering, single linear models have given way to multiple nonlinear approaches. Unfortunately in spite of the increasing mathematical sophistication and higher computing capabilities, problem complexity is still a hurdle to model adequacy. Consequently, other non-traditional approaches to modelling and problem solving like qualitative modelling, neural networks, fuzzy logic, expert systems and probabilistic reasoning have been explored by various scientists and engineers

(Johansen and Murray-Smith, 1997). The divide-and-conquer approach to modelling complex systems has been employed for many years. In this case the problem is decomposed into a number of simpler sub-problems which are then solved independently. The solutions of the individual problems become the solution of the original complex problem. Its success is however dependent on finding the right axes for decomposition.

A complex system can be decomposed along 5 axes – physical components, phenomena, mathematical series expansions, operating regimes and goals (Johansen and Murray-Smith, 1997). This new methodology - the operating regime approach - partitions the range of the complex system into multiple models or operating regimes where simpler functions can be employed to represent the relationships we want to model. One has to be aware of the trade-off between the size of the operating regimes and the complexity of the local models. At one extreme, a model can be represented by one large operating regime that covers the entire range of operation. In this case the local model is complex and equivalent to the global model. The operating regime on the other hand can be divided into such smaller units in which the local functions are constant values. Polynomial local models of arbitrarily low order (e.g. linear or constant local approximation) on a compact domain can be used to approximate any continuous function to an arbitrary uniform accuracy (Johansen and Murray-Smith, 1997; Kosko, 1994 and Singh, 1994).

The operating system approach is very useful in modelling nonlinear dynamic systems using computationally intensive data-driven techniques (Johansen and Murray-Smith, 1997). The method is knowledge-driven and integrates experimental data with the user's knowledge about the system. The processes essential to this method are experimental design and data acquisition, raw data processing and analysis, analysis of a priori knowledge including physical laws and available models, machine modelling (structure and parameter identification), model reduction and simplification and model validation, analysis and interpretation (Johansen and Murray-Smith, 1997).

Generally, an experimental process is iterative and the results of experimentation can offer valuable insights into the improvement and understanding of a system's response. However one can obtain good experimental results by using efficient methods of experimental design and good data modelling and analysis techniques.

6.3.1 Experimental Design Framework and Model Correctness

The problem solving methodology employed in this framework consists of a solution paradigm, expected solution criteria and assumptions. In order to obtain reliable experimental results, the paradigm employed in solving the problem must be able to capture the true definition or nature of the problem, the data involved, some basic data structures, file management issues, tasks and subtasks involved in solving the problem as well as a generic control regime (Fensel and Motta, 1996). The expected solution criteria involve the methods employed in obtaining the solution. These methods may be limited to a single solution methodology or a number of solution methodologies each of which can be specifically employed in solving certain predetermined tasks. In this case specific methods are used for the various tasks. This approach appears to yield a better solution.

The underlying assumptions are based on prior knowledge and they cover the type and properties of the knowledge structure that are required to solve the problem. These assumptions are especially important for intractable and complex problems. The general problem solving methodology employed in this work is outlined in Figure 6.1. In general in a well-defined problem, the determination of an axis of decomposition is the starting point of the experimental design process. This is then followed by an optimized experimental design process. This may be the single most difficult and time consuming element of the experimental design framework. The experimenter will basically have to deal with the question of what to do to get the right answers.

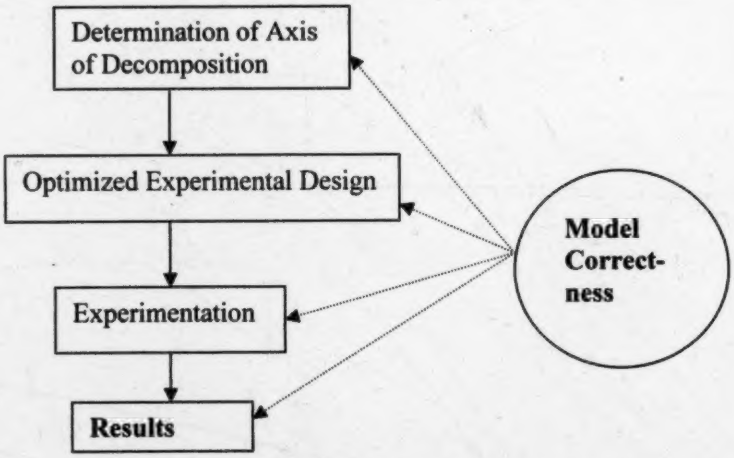

Determination of Axis of Decomposition
Optimized Experimental Design
Experimentation
Results
Model Correct-ness

Figure 6.1 Methodology for Designing Correct Experiments

There are several measures of model correctness such as lowest cost, quality, highest benefit/value, process-dependent, level of error, tolerance, and deviation from a benchmark. The measure of correctness applicable to each type of experimental model and various sections of the same experimental model may be different. The measure of the correctness is also dependent on the sub-processes driving the experiment. Model correctness should be reflected in every stage of the experimentation process.

6.3.2 Experimental Design for Optimized 3D Open Pit Mine

In this chapter, the models presented in Chapter 4 and computer algorithms in Chapter 5 are validated using data from an actual gold mine and a coal mine. Even though the gold mine models will involve both the design and optimization process, the coal mine models are limited to the block models. This is because coal mining follows a different process and the methods developed in this work cannot be directly applied to optimize it. In the case of the gold mine (an actual open pit mine), the two main experimental designs involved are (i) the design experiments and (ii) the optimization experiments.

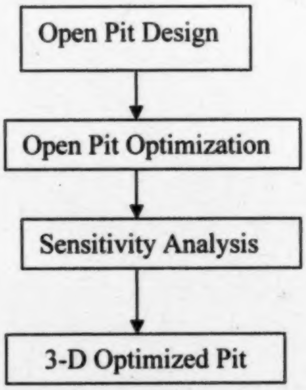


Figure 6.2 Design and Optimization Experiments

The new intelligent simulated annealing – neurogenetic and Lerchs-Grossmann algorithms are applied to 3-D models of the Sabi Gold Project to optimize the pit limits, evaluate its economic potential, and compare the optimized results. The mineral price model has already been validated by Achireko(1998) and will be employed directly in this work. Slope angle, and block value (ore grade) are the random variables studied in this experimentation as the net present value of an optimized pit (set of block) is dependent on them. The sequence of experimentation is outlined below in Figure 6.3.

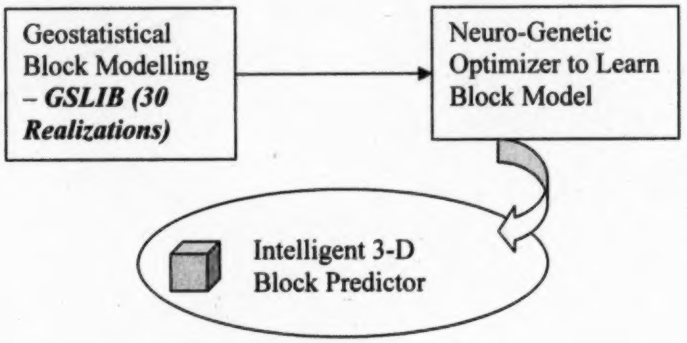

Geostatistical Block Modelling – ***GSLIB (30 Realizations)***
Neuro-Genetic Optimizer to Learn Block Model
Intelligent 3-D Block Predictor

Figure 6.3: Intelligent Block Design Paradigm (Gold Mine)

The intelligent 3-D open pit mine design and optimization experiments commence with geostatistical modelling and evaluation of the ore reserve (grade block modelling) and the use of artificial intelligence to design stable slopes. The slope is then converted to block equivalent. The selection of the block to be mined is done by using the 1 to 9 block convention. This 3-dimensional array of blocks is then employed in the optimization. Slope design experiments are also performed using adaptive logic network (ALN) and neuro-genetic optimizer (NGO). The two major processes involved in the block design are geostatistical block modelling with GSLIB followed by neuro-genetic predictor of block values. Parts of the rock data and block model are shown in Appendices 6.1, 6.2, and 6.3.

Two algorithms have been developed for the optimization, namely the incremental pit and the simulated annealing algorithms. However the simulated annealing algorithm is employed since it is a stochastic search process. As shown in Figure 6.3, the optimization process will involve some sensitivity analysis. During the sensitivity analysis a number of experiments will be run using the simulated annealing algorithm and the standard statistics will be analyzed.

In the case of the coal mine (an actual open cast or strip mine), the experimental designs involves only an aspect of the design phase and nothing on the optimization phase. This has everything to do with the nature of the open cast mining system itself and the way the block models were developed in Mincom. This will be further explained in Chapter 7. Part of the coal mine block data is shown in Appendix 6.4.

The framework for the entire optimization experiments is summarized in the flowchart shown in Figure 6.4. At the start of the optimization process, data from the simulated block models and the optimized pit slopes are input into the simulated annealing algorithm. The output from the first part of the optimization process is stored. If the

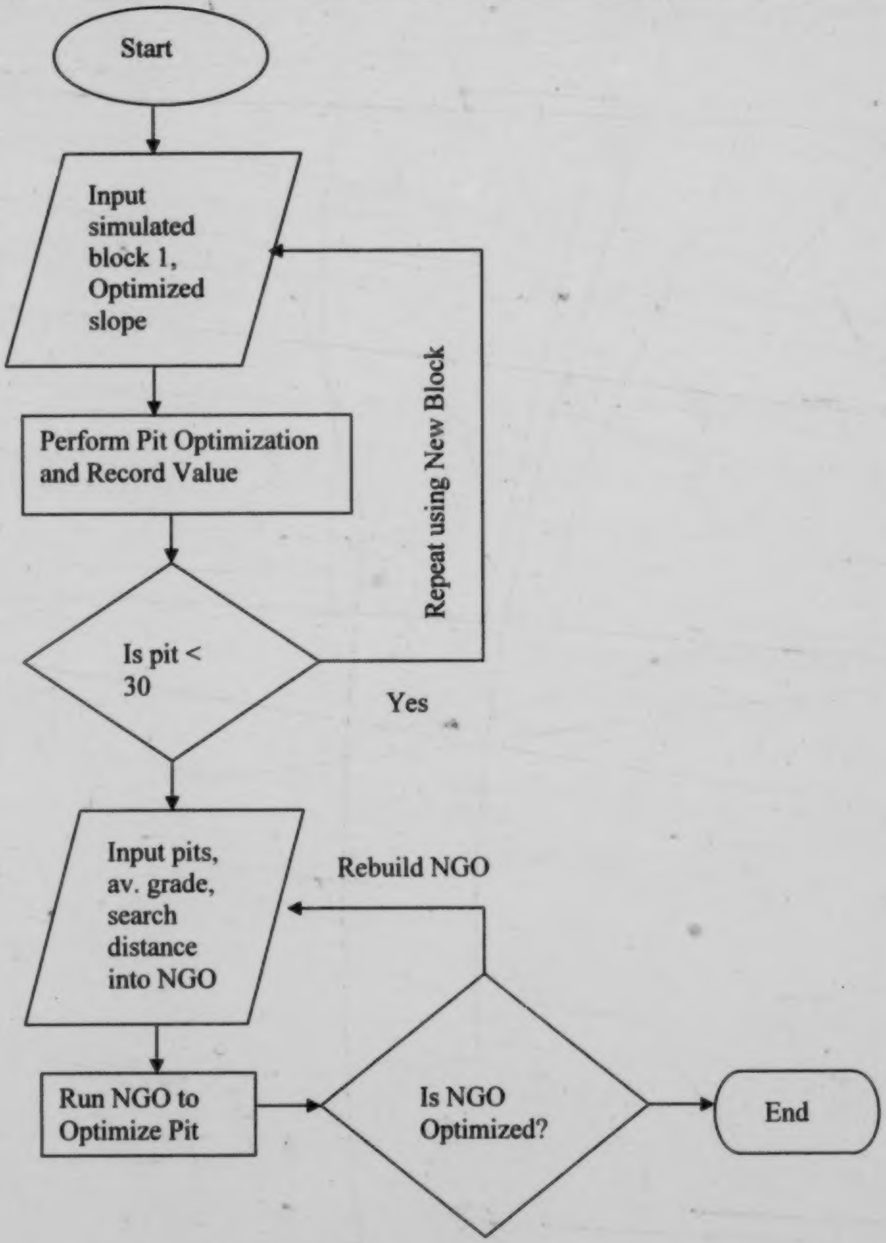

Start
Input simulated block 1, Optimized slope
Perform Pit Optimization and Record Value
Is pit < 30
Repeat using New Block
Yes
Input pits, av. grade, search distance into NGO
Rebuild NGO
Run NGO to Optimize Pit
Is NGO Optimized?
End

Figure 6.4 Flowchart for the Pit Optimization Process Using Simulated Annealing - Neurogenetic Optimizer

number of experiments in this phase is less than 30, more experiments are run. Altogether about 33 experiments were run. The objective of each of the experiments is to find the combination of blocks, which optimizes the open pit mine. The input data are the simulated block values from the GSLIB program and the optimized pit slope. The block data is first input into D45BLK.C program (Appendix 6.5) which outputs vital statistics as the number of blocks, average grade total value of all the blocks, the maximum grade value and the minimum grade values. The 3D block program SIM3D.C (Appendix 6.6) is then activated which optimizes the pit and output the optimized pit value.

In the second phase of the optimization process, pit values together with the average grade and the search distance are input into the neurogenetic optimizer. The optimizer is run and the output is checked to see whether it meets requirements. If it does meet requirements the process is ended. If it does not meet requirements the NGO building process is repeated. The optimized or best NGO has the minimum deviation between the predicted and actual values.

6.3.3 Experimental Design for Optimized Pit Slopes

Due to the complex field properties, experimental design for the pit slope is aimed at using intelligent algorithms to find the best pit slope. Slope walls define the amount of extra waste material that will have to be excavated to gain access to ore. A lower slope angle may reduce the risk of slope failure and increase the amount of overburden to be excavated. However a higher slope angle can increase the slope failure risk and reduce the total overburden material to be excavated. Determination of the right slope angle is therefore important to the economics of the entire mining operation. The slope angle experiments were done using artificial intelligence (neuro-genetic algorithm).

6.4 Block Modelling Experiments Using Geostatistics

There are two main aspects of the open pit design experiments, namely the geostatistical modelling and evaluation of the ore deposit which will result in a 3-D block model and the slope design. In this section the geostatistical modelling processes and software are described. Figure 6.5 depicts the entire geostatistical modelling and evaluation processes, which are carried out using the Geostatistical Software Library (GSLIB) package.

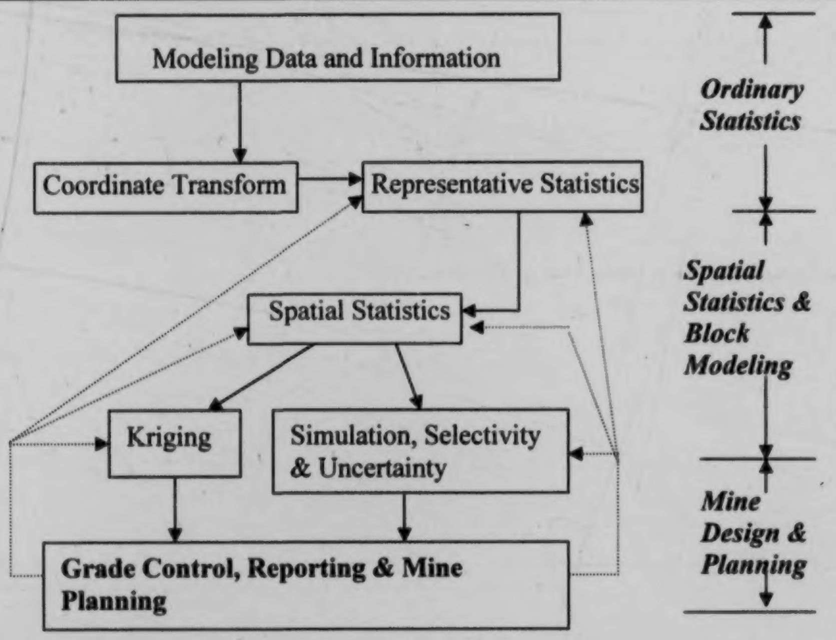

Modeling Data and Information
Coordinate Transform
Representative Statistics
Spatial Statistics
Kriging
Simulation, Selectivity & Uncertainty
Grade Control, Reporting & Mine Planning
Ordinary Statistics
Spatial Statistics & Block Modeling
Mine Design & Planning

Figure 6.5: Sequence of Actions in Reserve Modeling and Evaluation

The executables to be employed in the initial part of the experimentation are LOCMAP, HISTPLT, PROBPLT, HISTSMTH, DECLUS, SCATPLT and DRAW. The programs to be used in the variogram modelling are GAMV and GAM, VARMAP, NSCORE and BIGAUS. The KT3D and SASIM programs are used in the kriging and simulation stages of the experimentation. Other programs to be used are AFFINE, BLKAVG, GAMMABAR, and GTCURVE (Deutsch, 1998).

6.4.1 Location

The Star Consolidated Gold Mine a subsidiary of Zimbabwe Mining Development (ZMDC) is located 10 kilometers south-east of Zvishavane in the midlands province of Zimbabwe (Frimpong, 1988).

6.4.2 Description of Input Data

The Star Gold Mine is created from a variant of the actual gold mine in Zimbabwe. This project contained an oxidized outcropping gold deposit. Feasibility studies in 1988 established open pit mining and leaching technology for extracting the deposit. The data used in the block modelling are assayed gold grade values from the Star Project open pit mine. These grade values are estimates of the overall compositions of mining blocks, each obtained from a composite of the blast-hole samples collected within a block. They are used as known samples to find the values of other gridded blocks. The data set of the Star Gold project which as employed in the analysis contains 5624 gold data points. The mean of the gold values is 1.62 g/t. The minimum gold grade is 0.00 g/t and the maximum value is 3.86 g/t. Part of the data is displayed in Appendices 6.1 and 6.2. It consists of x, y, z coordinates, rock types and grade values.

6.4.3 Geology and Background Information

The deposit is situated within the basement gneiss in the Chikuraukwe Ridge along the North-South fault (Frimpong, 1988). The outcropping mineralized body is essentially contained within two sub-parallel, almost vertical shears varying from about 50 to 100 meter apart. The orebody extends for more than 2 km with a north-ward trend, mainly at or near the crests of the ridge. The gold is made up of 20-25 % free gold and the rest is associated with pyrites. Other associated minerals include chalcopyrite, bornite, calcite, epidote, magnesite, and the main gangue mineral is quartz. Past mining activities have been confined to auriferous pods aligned along stressed-relief fractures running at a few degrees east of north. These pods are discontinuous horizontally and vertically, pitching steeply to the north, and are highly siliceous. Average overburden thickness is about 0.5 m and the oxidized zone (with ore amenable to heap leaching) persists to a depth of about 30 m. The bank density of the ore is in the order of 2.76 tonnes/m^3, and the loose density is 1.700 tonnes/m^3.

The rock quality designation (RQD) is about 40 %. Metallurgical tests of the ore has indicated that the deposit is amenable to heap leaching. Actual ore processing have yielded recoveries of between 60-76% and further tests are underway to improve the recovery.

6.5 Block Modelling

The block modelling experimentation which is aimed at modelling and evaluating the ore reserve, is a multistage process. As depicted in Figure 6.5, the entire process can be divided into ordinary/classical statistical analysis, spatial statistics and block modelling, and mine design and planning. Built into the process are such things as checking for model correctness, change of support correction and kriging/simulation.

6.5.1 Machine Learning (Adaptive Logic Network) Experiment

The objective of the ALN training is to fit an optimized model to the slope data so that this optimized value can be employed in the pit optimization model. The process outlined below is used to run a factor of safety experiment for the slope design and to investigate estimation of block values. Using a combination of the right architecture and coefficients, an ALN can be a powerful tool.

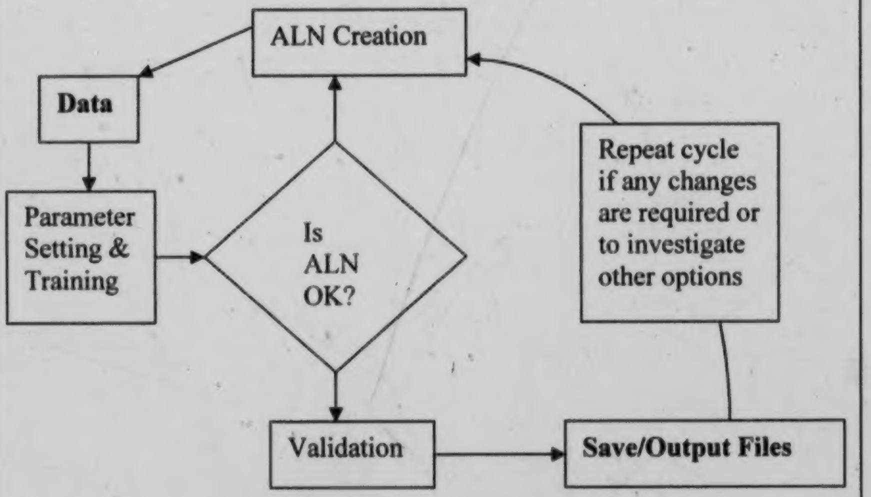

ALN Creation
Data
Parameter Setting & Training
Is ALN OK?
Repeat cycle if any changes are required or to investigate other options
Validation
Save/Output Files

Figure 6.6: ALN Experimentation Processes

The ALN software employs an iterative least squares technique in reducing the root-mean-square error. To use the ALN, a root mean square error tolerance is stated. The ALN then starts with a single linear piece, which is equivalent to a standard multiple regression.

The single weight vector coefficients are estimated to minimize the root-mean square error. If the calculated root-mean square error is larger than the error tolerance, the single piece breaks into two pieces which are either AND (maximum) or OR (minimum) functions. If any of these new pieces is responsible for a prediction error, its coefficient vector is altered to enable the new piece adapt. The process ceases when the root-mean square error is lower than the threshold value for all the pieces. The entire process is represented in Figure 6.6.

Thus the steps involved in the creation of an ALN include,

(i) creation of the correct ALN

(ii) putting training and test data into the ALN

(iii) ALN training

(iv) ALN testing

(v) ALN validation

(vi) Saving and ending

The initial step in the experimentation process is the creation of an ALN of appropriate geometry and dimension. The dimension of the ALN should match the number of variables in the problem to be solved. There are 2 types of geometry for ALNs, namely fixed and growable. A fixed geometry has a given number of layers and is used in situations where we know that the particular geometry fits the data perfectly. The default mode - a growable tree, commences with a single linear piece and expands automatically to fit the data.

The training and test data can be loaded into the model via the file-import format, keyboard entry or through cut and paste from a spreadsheet or data file. If the data is loaded directly from another file, one must ensure that it is tab-delimited. Parameters for the loaded data are set prior to training. Optimal results can only be achieved by setting these parameters correctly. There are 2 settings – parameters for each variable and parameters for the entire tree. These training parameters which are error tolerances, weight and value bounds are set for each variable. In the advanced mode, the following parameters are set within the entire tree - learning rate, number of epochs and the root mean square error (RMSE). The ALN is trained thereafter (Dendronic Decisions, 2000).

When training stops, the various text files are examined to determine if the ALN so achieved is appropriate. If the answer is in the affirmative, the new ALN is then validated with a third set of data. Files can then be saved and the program ended. If the ALN is found wanting, the ALN creation process is repeated. At this stage the parameters are changed to achieve better results. The cycle can also be repeated to investigate other options.

6.6 Grade and Block Modelling with Neuro-Genetic Algorithm

In this section the neurogenetic optimizer is employed to model the block values. Grade values are not only required for exploration decisions but are also the basis of block models in mine design. It is therefore necessary to be able to obtain accurate values during interpolation and extrapolation, based on the model. Even though most earth science data, including ore grade are lognormal, they are transformed into normal distributions in order to be able to subject them to gaussian statistical techniques. Even though these are linear transformations, the underlying phenomena may not be linear. The use of gaussian techniques will therefore introduce errors into the model and may lead to inaccurate results in some instances. This may be especially true when dealing with multiple variables, for example a lead-zinc-copper deposit. This may also be true in gold deposits where the underlying variability is so pronounced that a simple variogram model can not fully capture it. In this work, a stochastic method – a neuro-genetic algorithm is used to model the grade values. The input data consist of x-, y- and z-coordinates and grade values of the gold deposit.

Block models developed using geostatistical software can not be used off-line to predict block values in an advancing pit. The entire process of changing the parameters and generating the blocks again to make operational decisions can be long and tedious. The data employed are the coordinates of the block and the block values. The data is imported into the neuro-genetic optimizer (NGO) [BioComp Systems Incorporated, 1995]. Data can be entered manually or brought into the software from a text file. Once the data has been loaded, the application is configured to build an optimized model. The NGO runs in two modes – the standard and the optimized. In the standard mode, one can build neural networks of their own choice. In the optimizing mode, however the capabilities of the NGO are called into play to search and select data inputs and perform genetic engineering of neural network structures. Data is automatically scaled using either logistic sigmoid, tanh or linear functions, depending on whether the data is exclusively positive or positive and negative. Data can be split proportionately or disproportionately into test-train-validation sets. Data can also be selected randomly and placed in the test set with reference to the proportion you specify.

The network structure, hidden neuron and output neuron parameters of the neural network can be adjusted. The network architectures available are back propagation, continuous adaptive time neural network, time delay neural network, probabilistic neural network, generalized regression neural network, self organizing map and temporal self organizing map. The genetic algorithm in the NGO generates a host of possible neural network structures. It then searches for and optimizes the best neural network in this pool of neural networks. Each neural network in the population is trained, the best networks are selected and their attributes are recombined to create a new population of networks. This cycle continues until some stopping criteria is reached. This criteria is either the root-mean square error, the number of cycles or the length of time (BioComp, 2000). The software displays the best network and about 5 to eight other ones, numbered from second best to the least.

6.7 Open Pit Slope Design Experiments with Neuro-Genetic Algorithm

As depicted in equation 3.8, the calculation of the factor of safety involves a number of parameters. Most of these parameters have an underlying variability, which cannot be captured in a single factor of safety calculation. The neuro-genetic software that is being used however has an input limit of 5 variables.

The data is therefore pre-processed by calculating the disturbing, resisting forces as well as the factor of safety for the various scenarios. The process for calculating the optimum factor of safety and building a predictive model for the open pit mine slope is depicted in Figure 6.7. In Figure 6.7, the rock data is employed in calculating the disturbing and resisting forces.

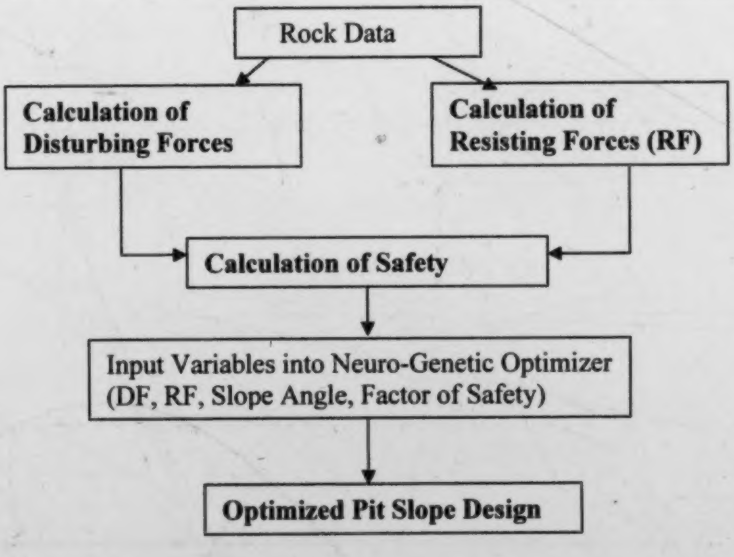

Rock Data
Calculation of Disturbing Forces
Calculation of Resisting Forces (RF)
Calculation of Safety
Input Variables into Neuro-Genetic Optimizer (DF, RF, Slope Angle, Factor of Safety)
Optimized Pit Slope Design

Figure 6.7 Open Pit Slope Design with Neuro-Genetic Algorithm

The safety factor for the set of disturbing and resisting forces is calculated as the ratio of resisting to disturbing forces. The disturbing force (DF), resisting force (RF), the slope angle and factor of safety are then employed as input into the neuro-genetic algorithm. The model is then run to obtain an optimized pit slope. There were 30 experiments run altogether using the various combinations of slope angle, factor of safety, disturbing and resisting forces.

6.8 Design of Optimization Experiments

As depicted in Figure 6.2, the open pit optimization process follows the design phase. The 3-D block model resulting from the design phase is employed in the optimization.

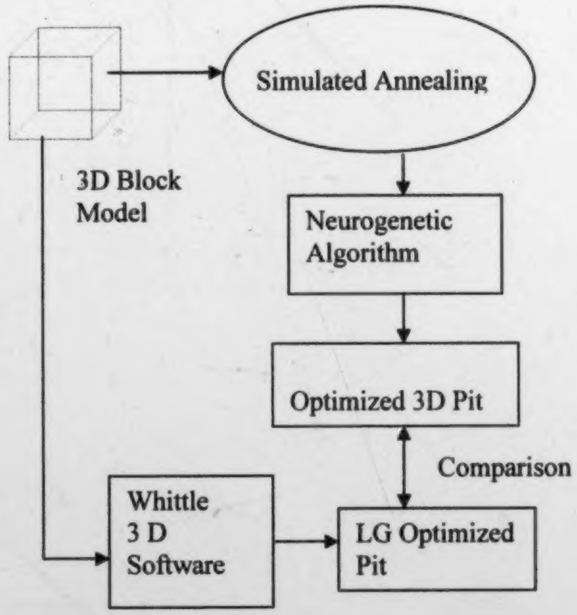

Simulated Annealing
3D Block Model
Neurogenetic Algorithm
Optimized 3D Pit
Comparison
Whittle 3 D Software
LG Optimized Pit

Figure 6.8 Design of Optimization Experiments

The three methodologies used in the optimization phase are simulated annealing, Lerchs-Grossmann and neurogenetic algorithms. Figure 6.8 illustrates a diagrammatic representation of the intelligent simulated annealing-neurogenetic optimization process. The output from the simulated annealing algorithm is fed into the neurogenetic optimizer.

The optimized pits resulting from both routes are compared to accounts for the differences. The optimized pit of choice is then compared with the pit resulting from the Lerch-Grossmann's algorithm.

6.8.1 Optimization by Simulated Annealing

Stochastic or randomized optimization algorithms like simulated annealing are used when there is no closed form analytic solution and the solution space is quite large. Thus the simulated annealing algorithm is an effective means of accounting for the stochastic processes underlying the parameters employed in open pit optimization. The central element of the simulated annealing algorithm as described in Chapter 5 is that the point (pit value) is in continuous random walk or motion; searching for the combination of blocks that will yield the minimum (-maximum).

In the case of the open pit optimization, the problem can be stated as finding the pit - combination of blocks of ore and waste, which will optimize the total value of the mine. Thus the problem is;

$$Find \quad B_T = (B_1, B_2, \ldots\ldots\ldots, B_{k-1}) \qquad so\,as\,to$$

$$Max \quad P_k(B_T) \tag{6.2}$$

Subject to:

$$\sum_{i=1}^{k} B_i = N$$
$$B_i \geq 0$$
$$B_i \text{ integer } (i = 1,2,3,\ldots\ldots, k) \qquad 6.3$$

The algorithm commences with an initial non-optimal set of blocks (which may be randomly selected). The algorithm improves itself by selecting a new set of blocks. This is because more often than not the mine will either start at an outcrop or in overburden. The mine operator will have to make a number of excavations before recording some revenue. The algorithm compares the value of the function at the trial point to the initial value. A random decision is then made to determine whether to move to this new point. The probability of accepting the new point is based on the value of the objective function at this point (P_{k+1}) as compared with the value at the previous point (P_k).

The four stages involved in translating and implementing the simulated annealing algorithm/pseudo-code outlined in Chapters 4 and 5, in the experimental setups are,

1. The basic program algorithm
2. The setup algorithm
3. The anneal algorithm
4. The decision algorithm

To facilitate the use of the basic program algorithm in the experiments, the algorithm is further broken down into the following experimentation steps;

- create initial list and set needed variables;
- try up to 100 different temperatures;
- anneal the list at this temperature and return the number of successful alterations made
- if there were no improvements found, end the program
- reduce temperature by 10 percent points

In the experimental setup, the setup algorithm creates the initial list and sets the initial temperature.

In the third stage of the experimentation, the annealing algorithm anneals the list of blocks at one temperature to find the best solution. The steps involved are;

- setting the number of successive changes to zero;
- run up to a 100 trials at the current temperature
- select a section of the list randomly to deal with
- randomly decide to select the next set of blocks and check if previous blocks have been mined
- calculate the change in net value (energy)
- make a decision on whether to include these new blocks in those to be mined using the decision algorithm
- if the decision is made to add these set of blocks to the list to be mined, then go ahead and change the list
- if sufficient number of successes have been attained, break out of the loop so that the temperature can be lowered.

Thus the annealing schedule consists of an initial temperature, a decrement factor and a final temperature. In order for the annealing algorithm to accept all proposed transitions, the initial temperature must be high.

The decrement in the temperature is given by the geometric progression;

$$T_k = \alpha T_{k-1} \qquad 6.4$$

The annealing process is discontinued if the desired number of acceptances is not achieved after 3 successive temperatures. In these experiments the temperature is defined as the quotient of the energy and the number of items in the list.

The run limit is set as the product of 100 and the number of items in the list and the success limit is one-tenth of the run limit. The decision part of the algorithm decides whether to accept the proposed new list of blocks. The steps involved in the decision-making process in the experimental setup are;

- a new list should always be kept if it has lower energy than the old list
- using the Boltzmann algorithm to decide whether a higher energy list should be kept
- if all hell break loose (failure occurs), reject the list

A decrease in value is more likely to be accepted than an increase in value. The acceptance or rejection of a block is governed by Boltzmann's probability depicted in Figure 6.9.

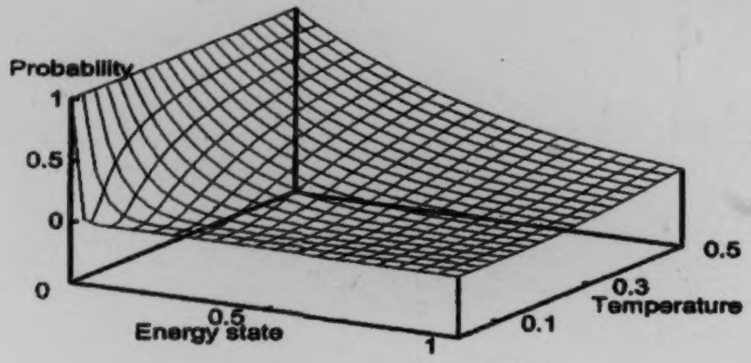


Figure 6.9 Boltzmann Probability Distribution of Energy States According to Temperature (After Spinellis and Papadopoulos, 2000)

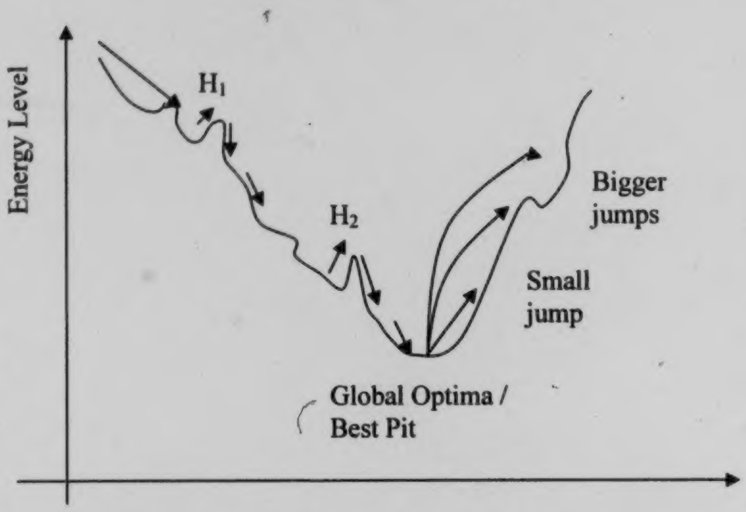

Energy Level
H₁
H₂
Bigger jumps
Small jump
Global Optima / Best Pit

Figure 6.10 Trajectory of a Simulated Annealing Experiment

The algorithm can wander uphill as well as downhill (see Figure 6.10). A higher energy path is not immediately rejected. Local or temporary higher energy paths – H_1 and H_2 are accepted or rejected by using the Boltzmann probability distribution criterion described above. Global optimum is reached when there is no further decline on the energy minimization. In order to ensure that a global optimum as opposed to a local optimum has been reached, the lowest energy is first stored. Then the algorithm is perturbed to see whether there will be a further deterioration in the energy (see Figure 6.10).

The results obtained from optimizing 30 different pits using the simulated annealing algorithm are fed into the neuro-genetic algorithm for true optimization of the open pit mine. Each of these block models have different grade values. The optimum pit value obtained from the new algorithm is compared with Lerchs-Grossmann and multilayer feed-forward algorithms. Finally the merits and demerits of the various optimization algorithms are discussed.

6.8.2 Risk/Sensitivity Analysis and Optimization by Neurogenetic Algorithm

The results from the simulated annealing experiments are fed into the neurogenetic optimizer for which will produce a trully optimized pit.

6.8.3 Optimization by Lerchs-Grossmann's Algorithm

Whittle's 3-D Lerchs-Grossman algorithm has been the standard industry-wide software for open pit optimization. The steps involved in using the Whittle 3D software can be summarized as follows pre-calculation of the slope using LGST.EXE file, 3D optimization using LG3D.EXE and examination of the results with LGPR.EXE (Whittle, 1990).

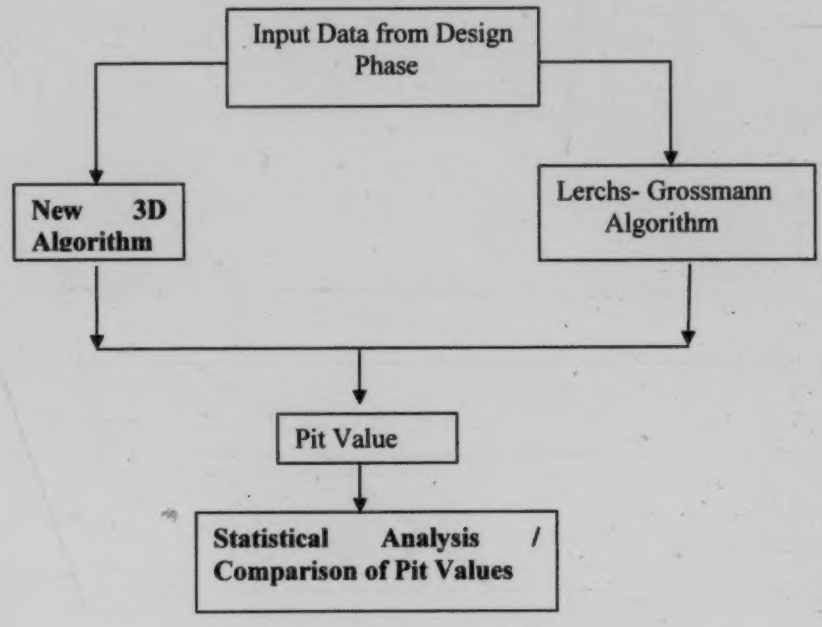

Input Data from Design Phase
New 3D Algorithm
Lerchs- Grossmann Algorithm
Pit Value
Statistical Analysis / Comparison of Pit Values

Figure 6.11 Comparison of Optimization Results

6.8.4 Comparative/Statistical Analysis of Open Pit Optimization Algorithms

The results from the 2 algorithms, namely, Lerchs-Grossmann and the simulated annealing-neurogenetic optimizer will be compared (see 6.11). This involves running 30 further experiments.

6.9 Conclusions

In summary, the design and optimization of an open pit mine involves a number of processes and algorithms. In order to obtain a numerical solution to the problem, the computer models are translated into actual code. The code is then employed in the running a series of experiments. Computer models are graphical and algorithmic representation of the solution to the problem at stake. The code is the set of written instructions in a specific programming language (C/C++ in this case), which the computer employs in solving the problem. The author wrote the program that generates a summary report from of the block data files. The simulated annealing program is a modification of the Algorithm of the Gods, which was originally written by Carlson (1997). In a situation where appropriate software exists like the AlnFit (Dendronic Decisions) and neurogenetic optimizer (BioComp), they are used in running the experiments.

CHAPTER 7

DISCUSSION OF RESULTS

This chapter constitutes a detailed discussion of the results obtained from the design and optimization experiments. It is quite evident from the literature survey and the work outlined so far that the parameters underlying the open pit design and optimization processes are stochastic in nature. The results discussed here were obtained from running experiments with the Alnfit, GSLIB, NGO, and SA-NGO codes and/or software.

7.1 Resource Modelling and Evaluation

Modelling and evaluation of a mineral resource is paramount to the design and optimization process and the subsequent planning and exploitation of the deposit so defined. The proper definition of the ore in terms of the quantity and grade of the underlying mineralization is at the heart of every mineral project. Bankers, analysts and mine operators are therefore very interested in the resource modelling method employed to outline the mineralization. Geostatistical ore reserve modelling and evaluation has taken center stage. Geostatistics in itself constitute a set of techniques and numerical methods aimed at modelling and evaluating regionalized (mineral resources) data. The methods range from statistical analysis through estimation to block modelling and simulation. The output from the various GSLIB programs are discussed in this particular section.

7.1.1 Exploratory Data (Statistical) Analysis

Mineralization was not uniform throughout the Sabi gold deposit and so it was necessary to categorize the data under different rock types. Even though all the various rock type models were finally put together to constitute the final model, the statistics from one of the rock types is presented here. Figure 7.1 depicts the histogram of rock type 2. It is quite evident from the histogram that the underlying distribution is lognormal. Almost all regionalized variables have a lognormal distribution and the presence of lognormality is important to the application of some of the techniques used in geostatistics (Deustch, 1998).

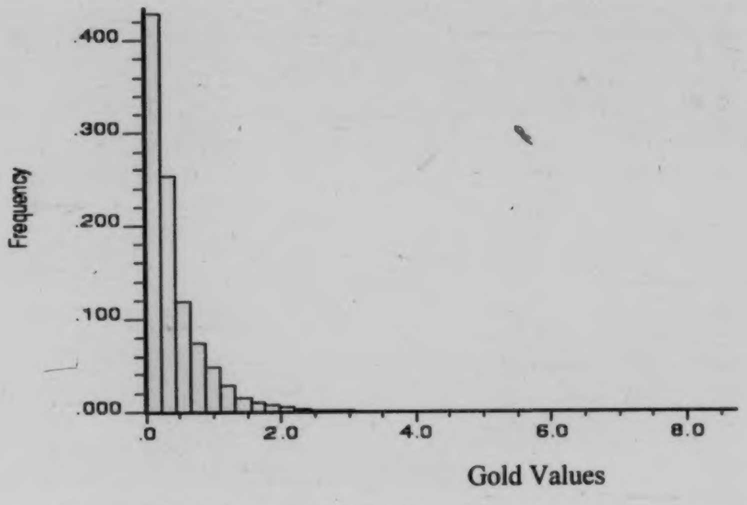


Figure 7.1: Histogram of Rock Type 2

A cumulative probability plot of the same data is shown in Figure 7.2. It also depicts a non-linear and non-uniform data. The data is declustered and normal scored. This makes it possible to apply the central limit theorem and other nice properties of the normal distribution to the data.

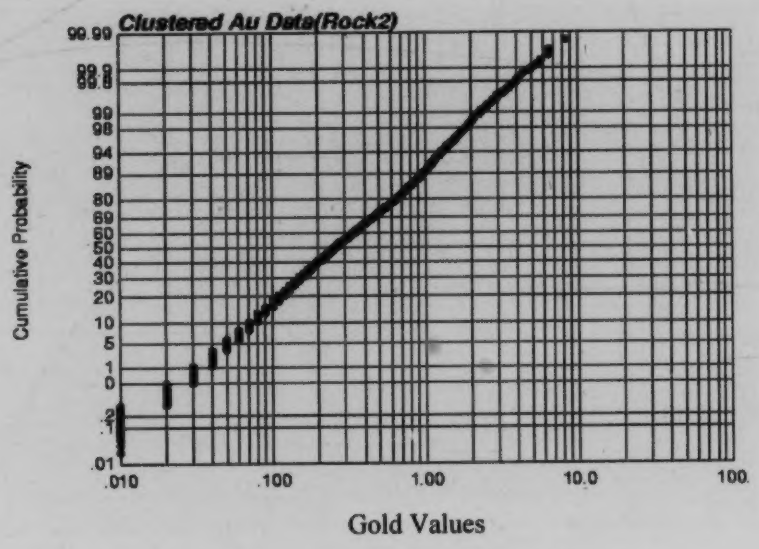


Figure 7.2: Cumulative Probability Plot of Rock Type 2

7.1.2 Variogram Modelling

In geostatistical modelling the variogram is the most important element. It is used to represent the relationship between euclidean and geological distance. Figure 7.3 is the normal scores variogram in the horizontal direction. Normally mineralizations are not the same in both the horizontal and vertical directions. The presence of anisotropy was also investigated. Variability could also be different for the horizontal and vertical directions. All the directional variograms for a particular direction follow the same pattern in each rock type, indicating the absence of anisotropy. However the variograms for the horizontal direction are different from those in the vertical direction (Figures 7.4 and 7.5).

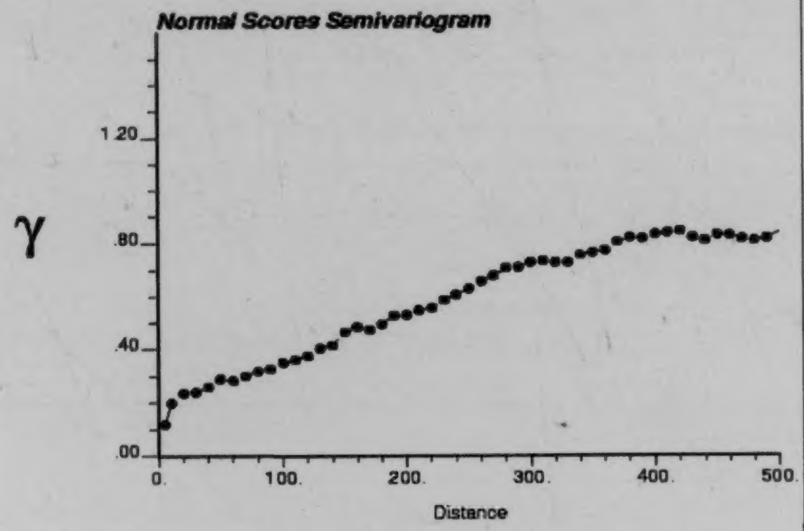


Figure 7.3 Normal Scores Variogram in Horizontal Direction

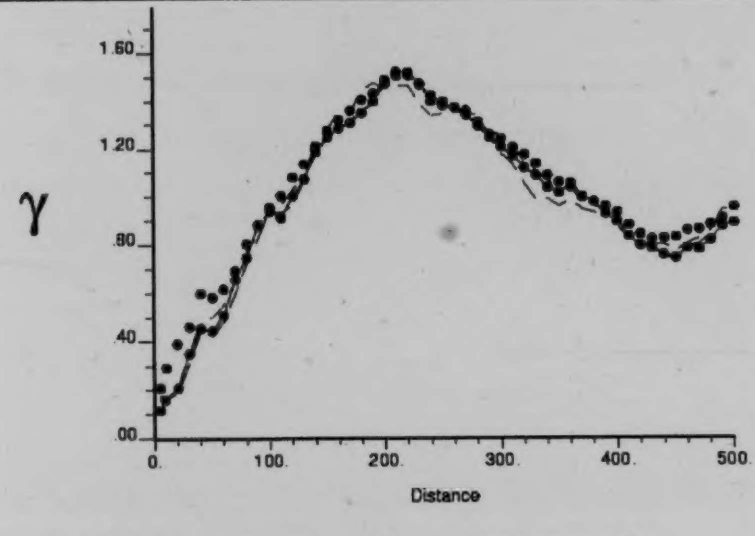


Figure 7.4 Horizontal Variograms in the N22.5^{0}E, N67.5^{0}E, N112.5^{0}E and N157.5^{0}E Directions.

The variograms fitted to the 2 rock types in the horizontal and vertical directions are of the form;

$$\gamma(h) = C_0 + C_l Sph\left(\frac{h}{a_i}\right) \qquad 7.1$$

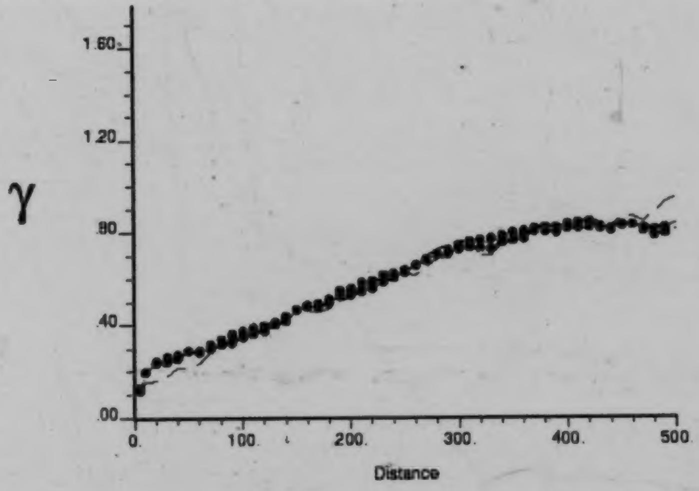


Figure 7.5 Vertical Variograms in the N22.5^{0}E, N67.5^{0}E, N112.5^{0}E and N157.5^{0}E Directions.

where γ(h) is the semivariogram of geological distance, C_0 is the nugget effect, C_l is a coefficient, h is the euclidean distance and a_i is the practical range or the distance within which the variogram value is 95% of the sill, in the horizontal or vertical direction. A summary of these parameters for two of the rock types are;

Parameter	**Rock Type 1**	**Rock Type 2**
C_0	0.1	0.05
C_1	0.9	0.95
$a_i = a_h$	105	130
$a_i = a_v$	105	500

The subscripts h and v represent the horizontal and vertical directions respectively.

7.1.3 Model Correctness

There is always some level of uncertainty associated with a variogram model even though it is considered as representative of the data employed in developing it. A variogram model must enable correct estimation of the phenomena or quantity of interest (grade, contaminant concentration) at the non-sampled locations within a reasonable degree of accuracy. The correctness of the variogram model is determined by a process termed cross-validation. The method involves removing one sample point at a time and using the neighboring samples to estimate the value of the "non-sampled" location. The actual error is the difference between the actual and estimated values. The expected or theoretical error is either the kriging variance or the kriging standard error calculated by the estimation. Generally the actual errors are averaged. This average error is zero for unbiased estimation and the ratio of the average variance and the average kriging variance is expected to be one.

Another method of comparison is to plot the kriged estimate against the true grade for a visual inspection of how well they compare. A correlation coefficient of 1.0 indicates that the model fits the data perfectly. The correlation coefficient for the various rock types averages about 0.87. This means that the experimental model can account for 87% of the variability in the gold deposit. The part of variability that can not be accounted for may be due to just sheer lack of information. The information effect could be more pronounced in precious metal vein-type deposits, intercalated by complex stockworks. The information effect can also be due to the data set and number of samples.

7.1.4 Kriging

Kriging is an estimation technique that is used for spatially distributed data. It is considered a generalized linear regression method that is used to estimate a continuous attribute. The results of running the kriging programme at radii of 45m, 60m, 75m, 90m, 105m and 135m with various block sizes were compared. The mineralization trend seems to be the same for all the radii. The Q-Q plot of the kriged and the actual grade values approximately fell on the 45° line thus indicating that the estimation process honoured the actual grades. The kriging estimation variance can be affected by errors in the calculation of the variogram. Any errors in the data set, stationarity assumptions, sampling and sample preparation results in high estimation variance. Negative weights can occur in kriging as a result of some samples being beyond the range or from shadow effects. A negative weight could even be your central value. Though some programs are known to eliminate negative weights, an action that increases the estimation variance, negative weights are not important if good and reliable estimates can be obtained in the presence of negative weights.

7.1.5 Simulation

Computer modeling and simulation of various processes have become acceptable forms of non-destructive testing, validation and extrapolation of diverse phenomena. The advantages of computer simulation can not be over-emphasized. During computer simulation, model parameters of systems are changed rapidly without jeopardizing system integrity. Theoretically, a sample point or grade is considered as a single realization of the regionalized random function. The class of random functions forming the basis of the phenomena under investigation has a characteristic distribution function and a variogram (covariance) model. The geostatistical model developed to date is one realization (albeit a smoothed version) of this random function in bounded space. Simulation can be used to draw realizations of the random function from the pool, which honor the model parameters including the variogram developed to date. In geostatistical modeling and evaluation of natural resources, such as the gold-copper deposit of this study, the reproduction of the presence or absence of spatial dependence of the one or several variables is paramount to the success of the modeling and evaluation process.

Gaussian-related simulation algorithms are applicable to continuous data. In Gaussian simulation model, the methodology adopted in this treatise, the z-covariance function is used to determine the spatial laws. Sequential gaussian simulation involves modeling and sampling one-point conditional distribution function at each of the nodes visited by the random sequence. The GSLIB subroutine SGSIM (Deutsch, 1992) was employed in the sequential simulation. The executable files could not handle the 3-D model. The main error message was that the available grid size is 150X150X1 but I have asked for 150X150X150 or 50X50X50, depending on the nx, ny and nz values entered in the parameter file. Changes were also made to the "include" file in order to run the program. The SGSIM.FOR (Deutsch, 1992) file was also edited, compiled, and run to build a new parameter file which was later altered and employed in the simulations. Fifty simulation runs were made for gold. The mineable tonnage was calculated to be about 15 million tonnes at a cut off grade of 0.9g/t.

7.1.6 Resources/Reserves Reporting:-

The results of this geostatistical analysis is the estimation of resources/reserves and classification into various categories. It is evident the entire in-situ mineral resource identified and estimated by geostatistical analysis cannot be mined because of variable mining and treatment costs and spatial variability inherent in the deposit. Besides geological in-situ resources, mineable or recoverable reserves according to Journel and Huijbregts (1978) depend on such factors as;

- selection criteria – maximization of the net present value of the entire operation is the predominant economic criteria. Other selection criteria can utility maximization, pareto optimality, net benefit maximization and cost minimization,
- cut-off grade or minimum mineable thickness - a grade above which a resource can be classified as ore can be used. Mineable thickness is an important issue in mining

thin high grade veins where access may only be gained through increase in mining thickness, for example in underground stopes.

- technological constraints - in a typical open pit environment, a block X can be mined if only all the blocks above defined by the cone of influence have been removed,
- support effects – size and geometry of the selection unit as compared to the support of the exploration unit (core samples),
- information available at the time of selection – selection may be based on kriged values with some level of uncertainty, whereas real values are recovered by the mill,
- geological boundaries – limits of various geological structures/rock types may affect the selection, and
- Regulatory requirements – regulations may define what can be classified as ore or waste.

The influence of support and level of information is expressed by the geostatistical variances of dispersion and estimation.

7.2 Adaptive Logic Network

Adaptive logic network is one of the fastest neural networks in application today. The ALN employs piecewise linear regression techniques to model complex non-linear systems. The training of a typical ALN consists of many linear pieces depending on the complexity of the function that is being fitted. The goal of training is obtaining a combination of linear pieces which fit the data with the lowest error. The error is obtained by summing the squares of the distances between the approximating surface and the output values. The output from the Alnfit is discussed in this section.

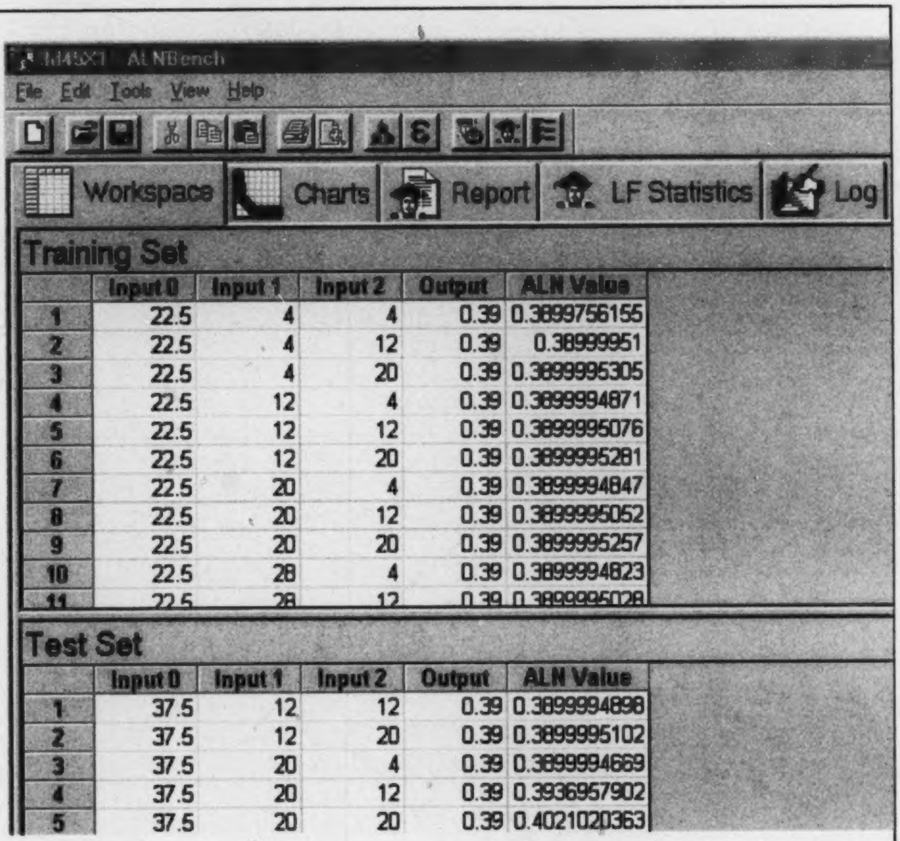
When dealing with real world data, which is complex and diverse in nature, there is concern for the tradeoff between precision and computational efficiency. ALNs use a supervised learning algorithm to learn patterns and complex relationships by using training set data, which consists of inputs and outputs. Supervised learning is equivalent to parameter estimation in statistics. In neural network terminology, an ALN can be regarded as some sort of feed-forward neural multi-layer perceptron. Linear threshold units are used in the first hidden layer. The other hidden layers as well as the output layer consist of logic gates of ANDs and ORs.

7.2.1 A Detailed Example of ALN Training or Estimation

In this work the data was assumed to have been generated from a function with unknown properties. It was assumed that the values had random noise in them. The data had been divided into two sets. Each data set was assumed to have come from the same function with the same type of noise. The noise is inferred from the two sets of data. The initial step was to train an ALN in standard mode with very small tolerance on the output variable. The value in the Smoothing Parameters window was set to 0.0001 and it disappeared when the "Epsilon" button was clicked. Then the Create ALN button was clicked to create a new ALN. The results have been displayed in Appendix 7.1. An extremely small root mean square error (RMSE) of 0.0035 was obtained on the training set, but a much larger one of 0.04 was obtained on the test set. This was characteristic of overtraining. In this case both the information and the noise have been fitted. Nevertheless, this indicates that the overfitted function has RMSE about 0.04. The data should never be any closer than that. This value was taken and divided by the square root of 2 to get an estimate of how far the test set is away from the real function. This yielded an output tolerance = 0.04 /1.414 = 0.028. Actually the noise was generated by adding a random value to the function that varied uniformly in -0.05 to 0.05. The correct value of the RMS error due to noise is thus 0.288. This value is very close to what has already been estimated.

The process is started over, but this time the output tolerance is set to 0.28. The allowable RMSE value on the training set should not be less than the output tolerance. If the output tolerance is set lower, the tree will grow very quickly and overtraining will result. The new results are presented in Appendix 7.2. According to this, there is still some overtraining, so increase the output tolerance to 0.07. The higher this is, the sooner the splitting of linear pieces will stop and the better generalization obtained. This process is continued until a good ALN is created.

Part of the ALN for the block values are shown in Figure 7.6. The filename appears in the uppermost left-hand corner. The first three columns (inputs 0, 1 and 2) are x-, y-, and z- coordinates of the blocks. Output is the block value. The ALN values during training and testing are shown in the final columns of the training and test sets. This screen shot depicts how the workspace appears after executing the model. As can be seen from this snapshot, even though the model fitted very well to the training set, there are some errors in the test set. Once again there are 4 variables involved. Note that the naming of the variables start from 0 to 3. In this case the dependent or reponse variable is the block grade (x_3) and the independent variables are x_0 (x coordinate), x_1 (y coordinate) and x_2 (z coordinate).



Training Set

	Input 0	Input 1	Input 2	Output	ALN Value
1	22.5	4	4	0.39	0.3899756155
2	22.5	4	12	0.39	0.38999951
3	22.5	4	20	0.39	0.3899995305
4	22.5	12	4	0.39	0.3899994871
5	22.5	12	12	0.39	0.3899995076
6	22.5	12	20	0.39	0.3899995281
7	22.5	20	4	0.39	0.3899994847
8	22.5	20	12	0.39	0.3899995052
9	22.5	20	20	0.39	0.3899995257
10	22.5	28	4	0.39	0.3899994823
11	22.5	28	12	0.39	0.3899995028

Test Set

	Input 0	Input 1	Input 2	Output	ALN Value
1	37.5	12	12	0.39	0.3899994898
2	37.5	12	20	0.39	0.3899995102
3	37.5	20	4	0.39	0.3899994669
4	37.5	20	12	0.39	0.3936957902
5	37.5	20	20	0.39	0.4021020363

Figure 7.6: A Screen shot of the ALN for the Block Values.

The ALN employed 312 linear forms to fit this model. The model involves 8 blocks and 15 decision trees (Appendix 7.3). It must be noted that the data shown in Figure 7.6 is only a snapshot of the actual data. Typically, an entire data set consists of about 5000 data points. If the x and y coordinates are the same the z coordinate is different, which indicates that each of the data set is different block of ore or waste. There are also charts and various statistics that describe the ALN model so developed. An example of a chart output by the model described above is shown in Figure 7.7.

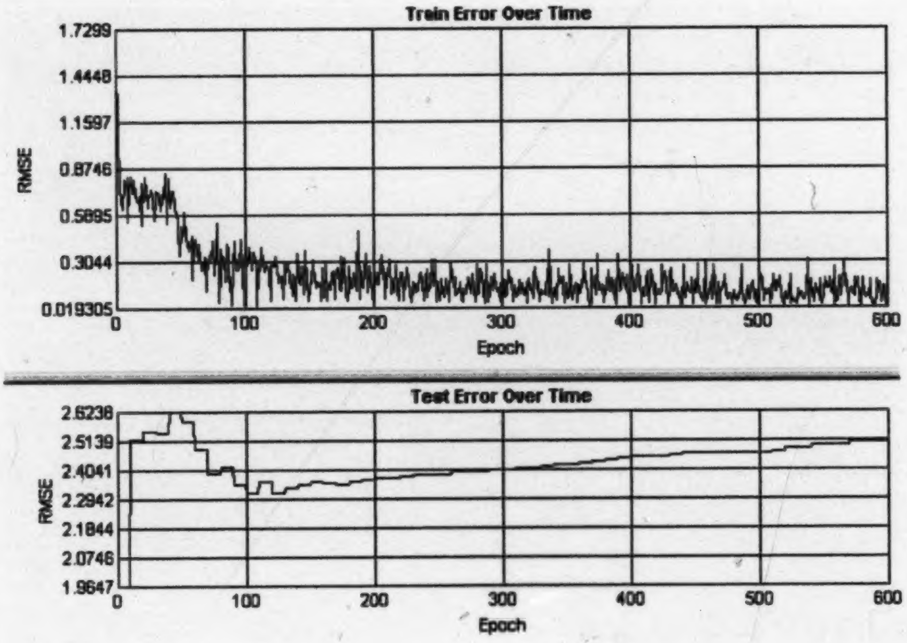


Figure 7.7: The Behavior of the Error Function During Training and Testing

This figure shows the behaviour of the training and the testing errors with respect to the number of iterations.

It is evident from these charts that the training error settled down rather quickly, just after 200 epochs. The training error settled at (a root mean square error) value of about 0.28. After this the training error oscillated around a mean value and did not change much even with increased iterations. The test error reduced to a value of about 2.3 and then rose again to a high of 2.5. This may be due to the differences in the test and training data sets and possibly overtraining. These issues were fully investigated but the training error did not improve much. Indications are that the ALN is sensitive to outliers and may not effectively model large discrepancies in the data. The variability may also represent the inability of adaptive logic networks to model gold mine grade data.

7.3 Neuro-Genetic Modeling of Block Values

Neural networks have the capacity to estimate non-linear functions and perform pattern recognition among other things. However developing an optimized neural network to suit a particular data can be a monumental task.

There could be hundreds and in some instances thousands of possibilities. With a complex combination of performance requirements such as learning speed, compactness, ability to generalize and noise-resistance together with growth in size and model complexity, a well-engineered solution as required by the block model, is a non-trivial task. The divide-and-conquer approach has therefore given way to model automation. Genetic algorithms, a stochastic optimization approach, have for good reasons become the methods of choice in optimizing neural network architectures. The genetically optimized neural network architecture employed in this work, is a problem-solving methodology that applies the rules of genetic reproduction, gene crossover, and mutation to a population of candidate neural network solutions. The operational cycle of a typical neuro-genetic optimizer as employed in this work is shown in Figure 7.8.

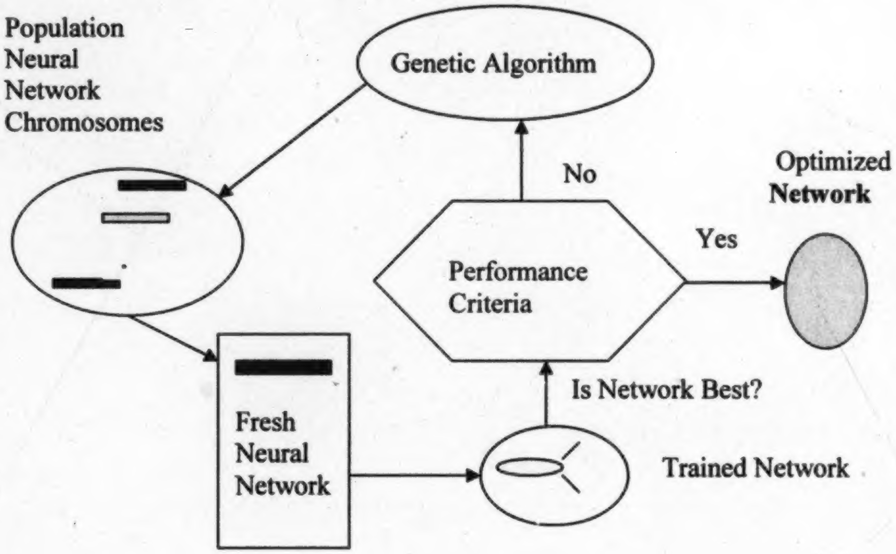
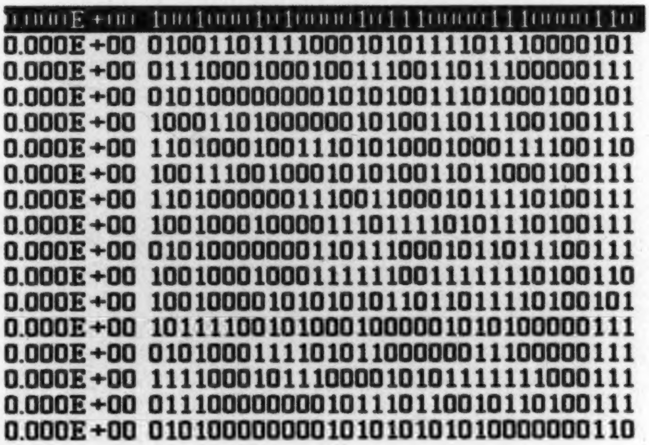

Population
Neural
Network
Chromosomes
Genetic Algorithm
No
Optimized
Network
Yes
Performance
Criteria
Is Network Best?
Fresh
Neural
Network
Trained Network

Figure 7.8 A Typical Neuro-Genetic Modeling Process

7.3.1 Modelling of Gold Block Values

The block modeling application was considered a classification problem. Using the features of the input data as a basis, the neuro-genetic optimizer learns to distinguish between 2 or more categories prevalent in the data. The network's highest neural output indicates the category or class the data record falls into. The upper limit of the accuracy is 100%, where all the records are properly classified. The first neural network model was evaluated by employing all the data available together with the maximum allowable hidden neurons. This use of all the available inputs may lead to the best neural network. This may however not be the case as it is evident in the block model. The software automatically employed the x-, y- and block values to obtain the maximum neural network. The predictions were completely out of order and only 40% of them were correct. The resulting model was also 2-dimensional and since the focus of this work is 3-D modelling, the 2-D model was not considered any further. This issue will be revisited later since it is quite interesting and may have resulted from the software methodology or some underlying phenomena in the data.

A portion of the population of genetically generated neural network architectures is depicted in Figure 7.9. The best neural network generated by the neurogenetic optimizer is on top of this list of candidate neural architectures. Using a probability of selection proportional to fitness, a set of parents is chosen in reproduction. Crossover mapping involves 2 parental or two offspring chromosomes. This process is random.



```
[illegible]
0.000E+00 0100110111100010101111011100001 01
0.000E+00 0111000100010011100110111000001 11
0.000E+00 0101000000001010100111010001001 01
0.000E+00 1000110100000010100110111001001 11
0.000E+00 1101000100111010100010001111001 10
0.000E+00 1001110010001010100110110001001 11
0.000E+00 1101000000111001100010111101001 11
0.000E+00 1001000100001110111101011101001 11
0.000E+00 0101000000011011100010110111001 11
0.000E+00 1001000100011111100111111101001 10
0.000E+00 1001000010101010110110111101001 01
0.000E+00 1011110010100010000010101000001 11
0.000E+00 0101000111101011000000111000001 11
0.000E+00 1111000101110000101011111110001 11
0.000E+00 0111000000001011101100101101001 11
0.000E+00 0101000000001010101010100000001 10
```

Figure 7.9 Portion of Genetic Population for Gold Block Model (with Back-Propagation Learning Algorithm)

Figure 7.10 shows 2 parents in the process of generating two offsprings. Note that the offsprings are not identical. No two of the genetic population in Figure 7.9 are therefore supposed to be identical. This results in a lot of different neural network options. Mutation as depicted in Figure 7.10, ensured diversity and prevented death (premature stalling or stopping).

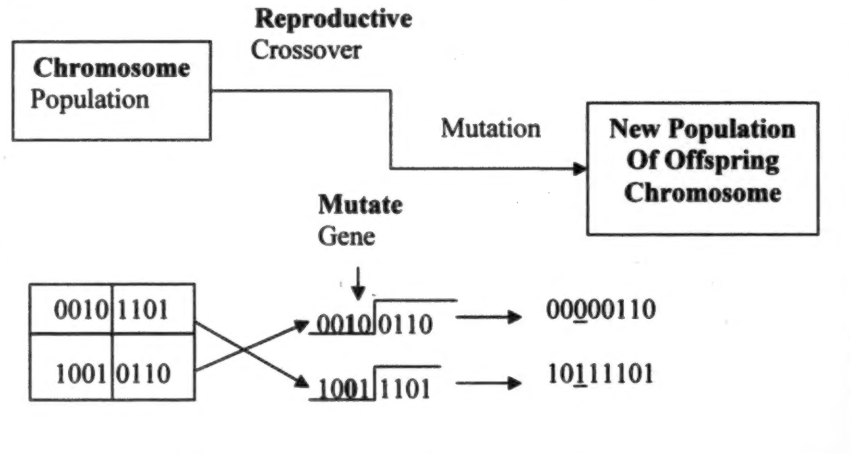

Reproductive
Crossover
Chromosome
Population
Mutation
New Population Of Offspring Chromosome
Mutate
Gene
0010
1101
1001
0110
0010
0110
00000110
1001
1101
10111101

Figure 7.10 Generation of New Population (After Looney, 1997)

A portion of the population of genetically generated neural network architectures is depicted in Figure 7.9. The best neural network generated by the neurogenetic optimizer is on top of this list of candidate neural architectures. Using a probability of selection proportional to fitness, a set of parents is chosen in reproduction. Crossover mapping involves 2 parental or two offspring chromosomes. This process is random. Figure 7.10 shows 2 parents in the process of generating two offsprings. Note that the offsprings are not identical. No two of the genetic population in Figure 7.9 are therefore supposed to be identical. This results in a lot of different neural network options. Mutation as depicted in

Figure 7.10, ensured diversity and prevented death (premature stalling or stopping). Reproduction, crossover, and mutation are the driving elements of the entire evolutionary process whereas fitness guides it.

The fitness measure affects which parents are chosen by the selection process to reproduce new population (offsprings). Fitness serves as a guide to the best or fittest population while mutation encourages diversity and thus prevents stalling. When the model was forced to employ the x-, y-, z- and block values, the NGO fitted another back-propagation neural network as the "best". The relationship between the desired and predicted values as given by the back-propagation model is depicted in Figure 7.11.

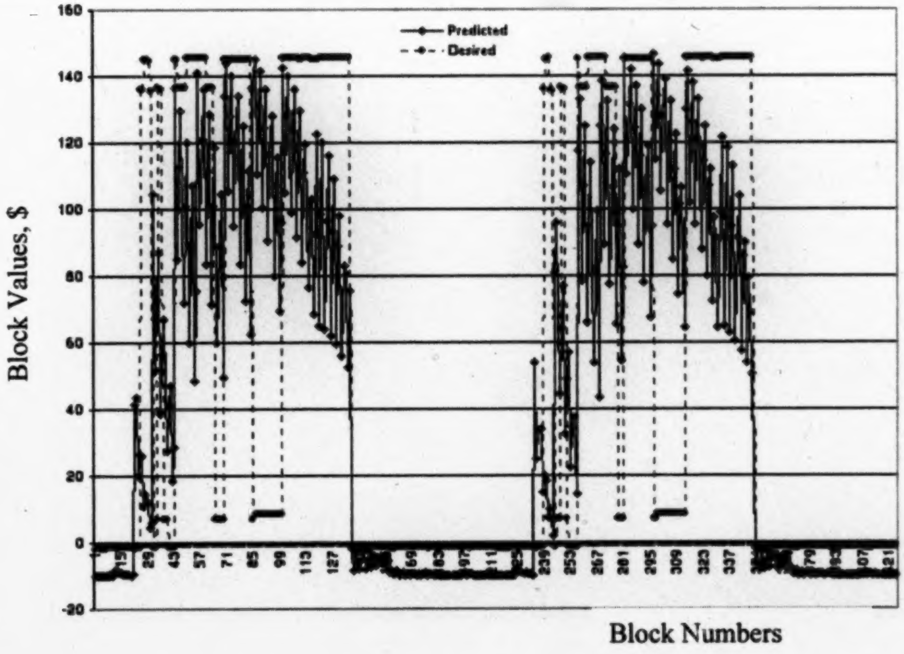


Figure 7.11: Desired and Predicted Values From Back-Propagation Model

The predicted values deviated substantially from the desired values. Even though the correlation coefficient as calculated in a Microsoft Excel (1997) spreadsheet was about 70%, it is evident that the deviations were substantial when they occurred. In some places the deviations were as much as 60 basis points (see Figure 7.11).

The f-statistic and t-statistic are 6.5×10^{-6} and 2.5808×10^{-5} respectively. In the case of the t-statistic, it was assumed that the data was paired.

A diagram representing the results of 30+ experiments using this model is shown in Figure 7.12. As can be seen from this diagram, the feedforward neural network (with backpropagation learning) could not predict the block values to an acceptable degree of accuracy. The experiments actually substantiated the problems with the back-propagation neural network. It was wrong more than 50% of the time. It cannot therefore be used a reliable estimate for mine planning. The output report for the back-propagation network is in Appendix 7.4. The best results were obtained by using a probabilistic approach. The 2 competing algorithms employed were the generalized regression neural network and the probabilistic neural network models to classify the data. The generalized regression neural network came up as the ultimate neural network for this application.

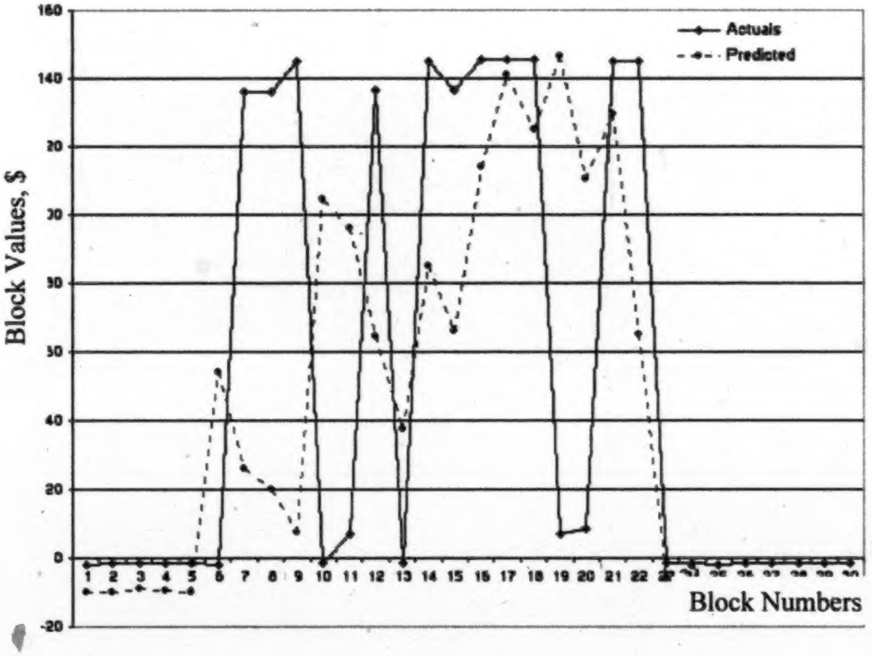


Figure 7.12: Graph of Experimental Versus Actual Results Using the Feedforward (Back-Propagation) Neural Network

As depicted in Figure 7.13, the predicted results highly correlates with the desired results. The correlation coefficient is 92%. The corresponding f-statistic and t-statistic are 5.31822 X 10^1 and 2.6785 X 10^{-2} respectively.

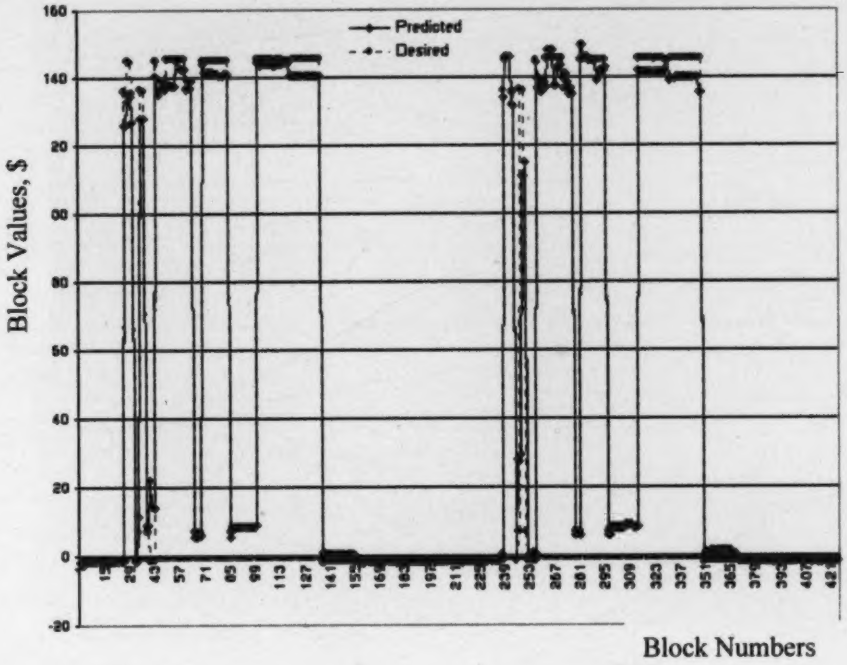


Figure 7.13: Desired Versus Predicted Values from GRNN

The results of experimenting with this particular generalized regression neural network (GRNN) are presented in Figure 7.14. The predicted values are mostly equivalent to the actual values. The only exception may be the extreme values. Generally the predicted values followed the trend of the actual values and can therefore be used as reliable estimates. The report for the best neural network model is a generalized regression neural network and it is outlined in Appendix 7.5.

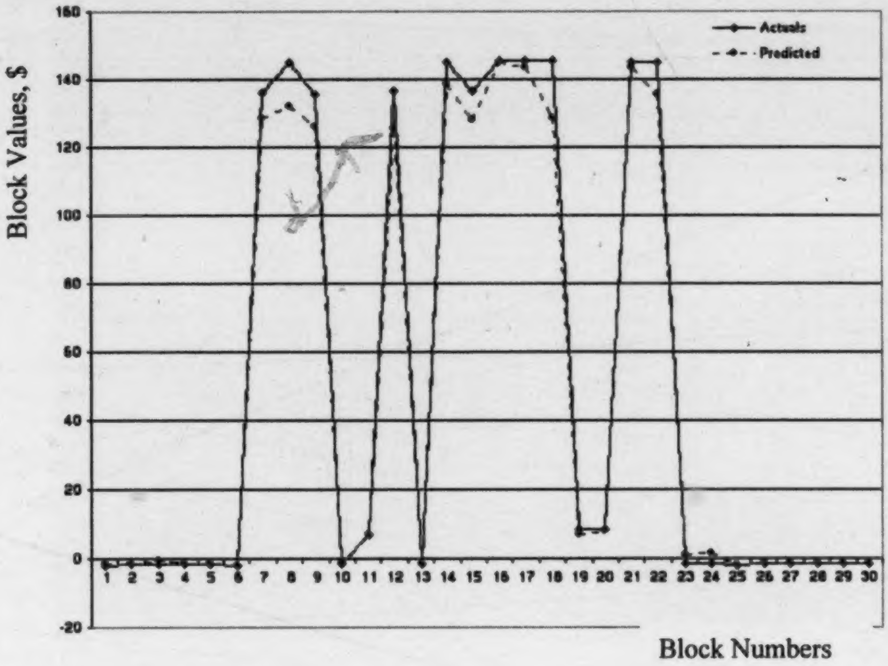


Figure 7.14: Graph of Experimental Versus Actual Results Using GRNN

Figure 7.15 is a graph showing the linear correlation between the actual block values and the values predicted using the neurogenetic optimizer. The network had two hidden layers with the first layer containing 213 neurons and the second layer, 2 neurons. The network so built was able to predict grade values at a considerably higher accuracy. The graph virtually falls on the 45^0 (one-to-one correspondence) line. This NGO predictor can therefore be effectively used to predict grade values in the pit.

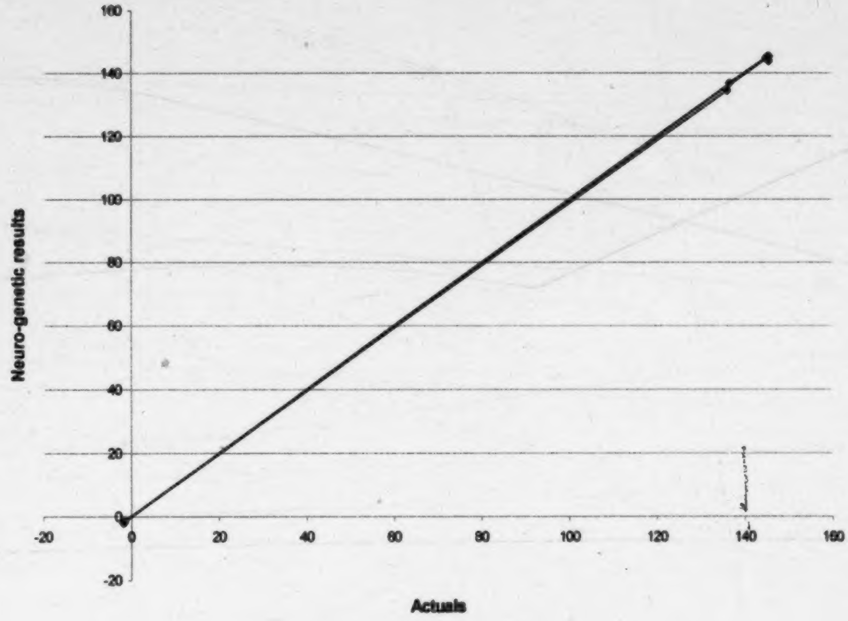
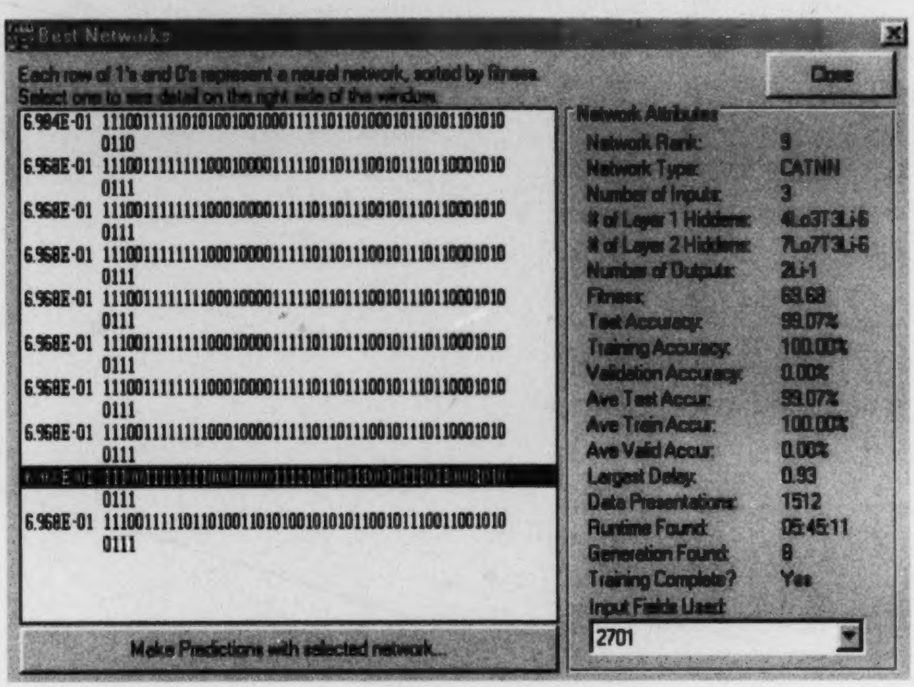


Figure 7.15: Predicted Versus Actual Block Values.

7.3.2 Results of Modelling of Coal Block Values

The results from the NGO software presented so far depict the performance of the algorithms in a gold ore block model. In this section the results of the coal block model experiments are presented. This time the NGO in the automatic mode generated a continuous adaptive time neural network (CATNN) as the best model to fit the data.

A graphical representation of the output is shown in Figure 7.16. The genetic algorithm representation of the candidate neural networks are coded by 0's and 1's. Thus each of the rows of 0's and 1's represent a particular neural network. These networks are sorted



Best Networks

Each row of 1's and 0's represent a neural network, sorted by fitness.
Select one to see detail on the right side of the window.

```
6.984E-01 11100111110101001001000111110110100010110101101010
          0110
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
6.968E-01 11100111111111000100001111101101110010111011000101010
          0111
[illegible]E-01 [illegible]
          0111
6.968E-01 11100111110110100110101001010101100101110011001010
          0111
```

Make Predictions with selected network...

Network Attributes

Network Rank:	9
Network Type:	CATNN
Number of Inputs:	3
# of Layer 1 Hiddens:	4Lo3T3Li-6
# of Layer 2 Hiddens:	7Lo7T3Li-6
Number of Outputs:	2Li-1
Fitness:	69.68
Test Accuracy:	99.07%
Training Accuracy:	100.00%
Validation Accuracy:	0.00%
Ave Test Accur:	99.07%
Ave Train Accur:	100.00%
Ave Valid Accur:	0.00%
Largest Delay:	0.93
Data Presentations:	1512
Runtime Found:	05:45:11
Generation Found:	8
Training Complete?	Yes

Input Fields Used:

2701

Figure 7.16: Continuous Adaptive Time Neural Network Model for Coal Block Model.

by fitness. The resulting network consisted of 3 inputs, 2 hidden layers and 3 output layers. This particular network is ranked number 9 and has fitness value of 69.68. One output layer for each seam. A summary of this network is in Appendix 7.5. The desired and predicted results from the network are presented in Figures 7.17 and 7.18.

There are a number of places in Figure 7.17, where there is considerable difference between the actual and predicted values. In the initial part of the modelling process the values predicted by the neural network were quite low when compared to the actual

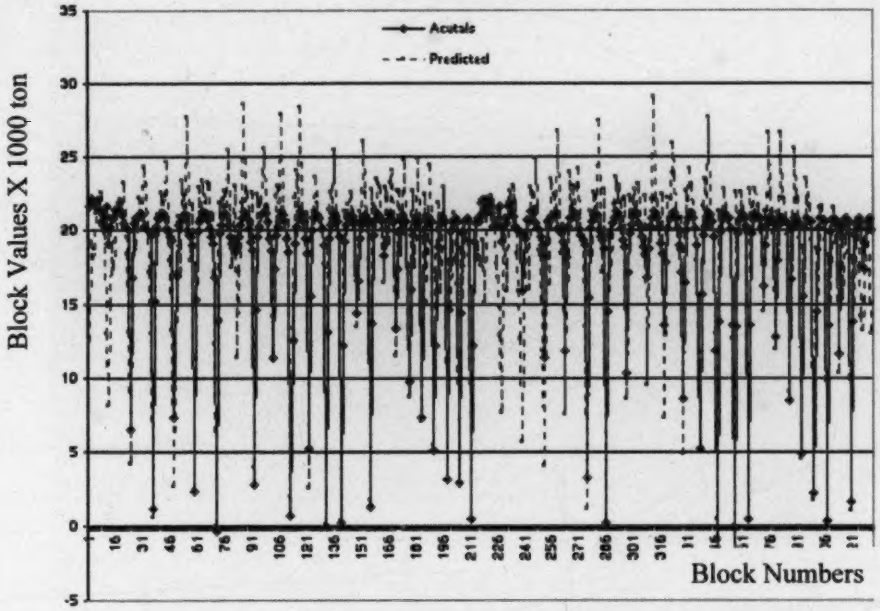


Figure 7.17: Desired and Predicted Results from CATNN – Coal Seam 1

values. The highest level of underestimation is in the initial and middle parts. The underestimation in these areas amounts to about 10,000 tons of coal. Most of the other deviations are overestimates. The highest level of overestimation is about 8,000 tons of coal. One half of the model almost qualifies as a mirror image of the other half. The overestimates are spread evenly in either half of the model.

The relationship between the desired and predicted results for seam 2 is very similar to that of seam 1(Figure 7.18). This means that the two seams are in the same depositional environment and might have been formed by the same depositional processes.

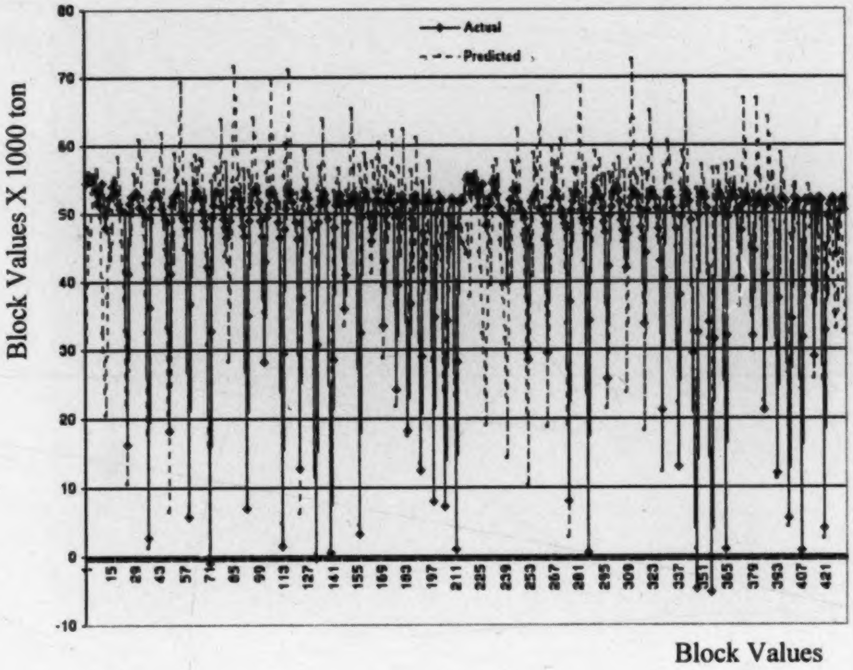


Figure 7.18: Desired and Predicted Results from CATNN – Coal Seam 2

However the level of overestimation is quite high. The underestimation at the beginning and mid-section of the model amounts to about 18,000 tons. This is due to the fact that seam 2 contains more coal than seam 1. The average amount of coal in seam 1 is about 18,725 tons, whereas the average in seam 2 is about 46,981 tons.

The results of running experiments using the continuous adaptive time neural network model for the 2 seams are presented in Figures 7.19 and 7.20 respectively.

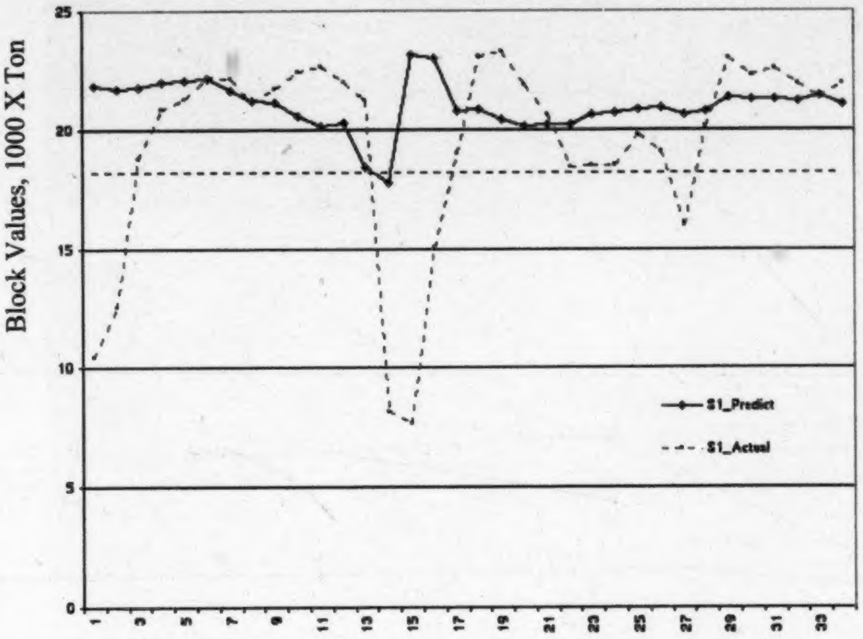


Figure 7.19: CATNN Experimental Results for Coal Block Model - Seam 1.

Its is quite evident from both Figures 7.19 and 7.20 that the predictions using CATNN model could not follow the actual trend of the block values. In the case of seams 1 (Figure 7.19), the difference between the predicted and actual values is about 15,000 tons of coal in the initial stages. This difference dropped during the course of the experiments and became equivalent at about 6 different places. The major discrepancies between the two models occurred in the mid-section of the block model. The difference in the actual and the predicted model (continuous adaptive time neural network) here amounts to about 18,000 tons of coal. This is equivalent to the average block value of seam 1, which is

18,725 tons. This big drop in the value of the blocks is localized and does not spread throughout the model.

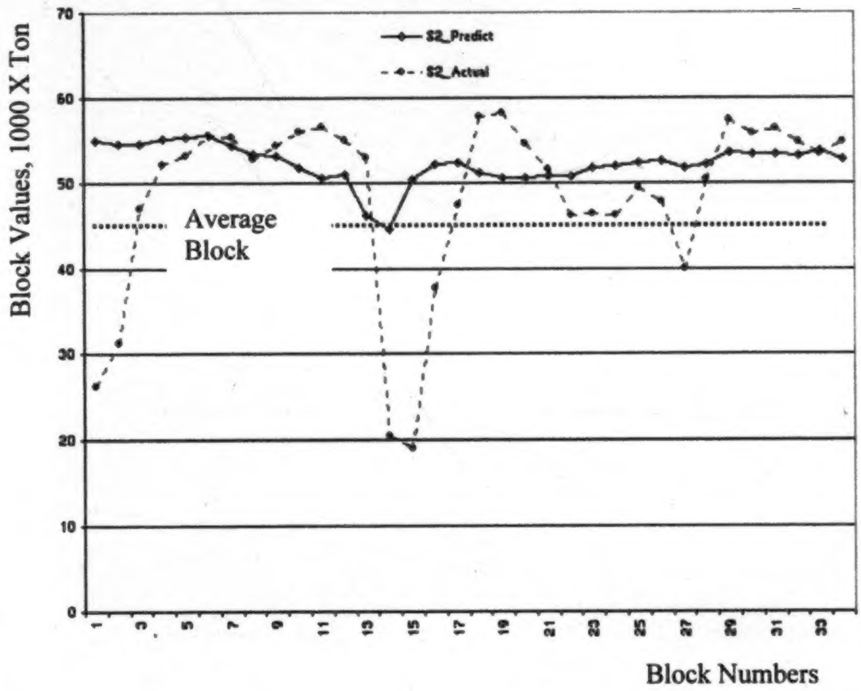


Figure 7.20: CATNN Experimental Results for Coal Block Model - Seam 2.

The sharp drop in block value may be due to a partial washout, an old coal fire or the location of an old river. The seam 1 model predicted rather higher block values (Figure 7.19). Generally the predicted values are higher than the actual values and almost all the predicted values are above the block average of seam 1.

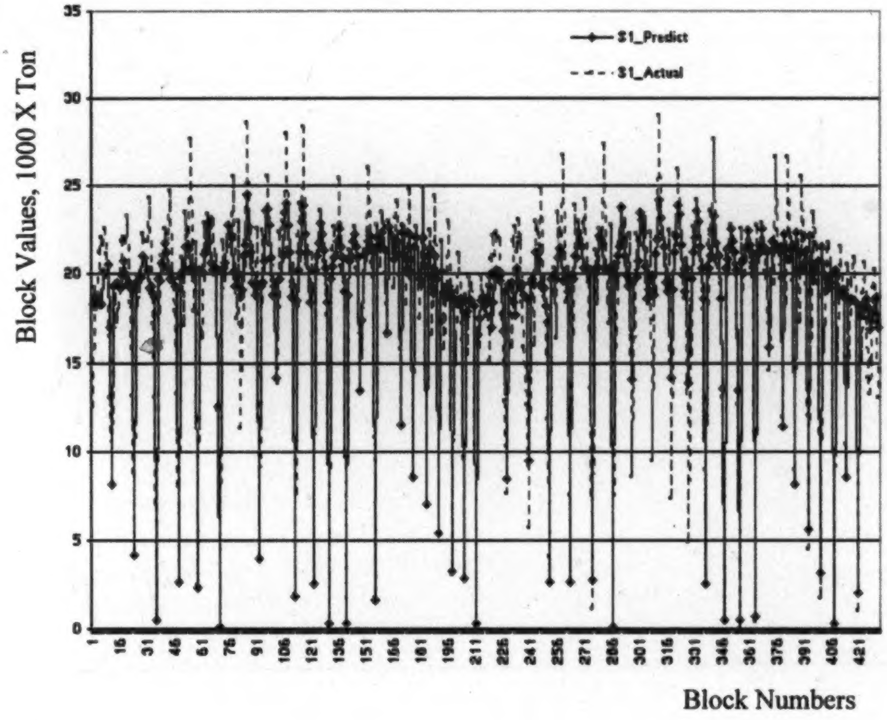


Figure 7.21: Desired and Predicted Results from GRNN - Coal Seam 1

In the initial experiments, the predicted value (55,000 tons) was twice the actual value (27,000 tons). The situation improved after two to three experiments. The actual and predicted values are equivalent in about 7 places on the graph (Figure 7.20). Once again the CATNN predictor could not pick the large depression in the block values in the about the middle of the experiments. The difference in value amounts to 33,000 tons. Though it is lower than the average block value in this case but it is still very high.

Predictions from the seam 2 continuous adaptive time neural network model are equally higher than the average values (see Figure 7.20). It is even more evident here that the CATNN models for the two coal seams could not be used to predict accurate values for mine planning/scheduling.

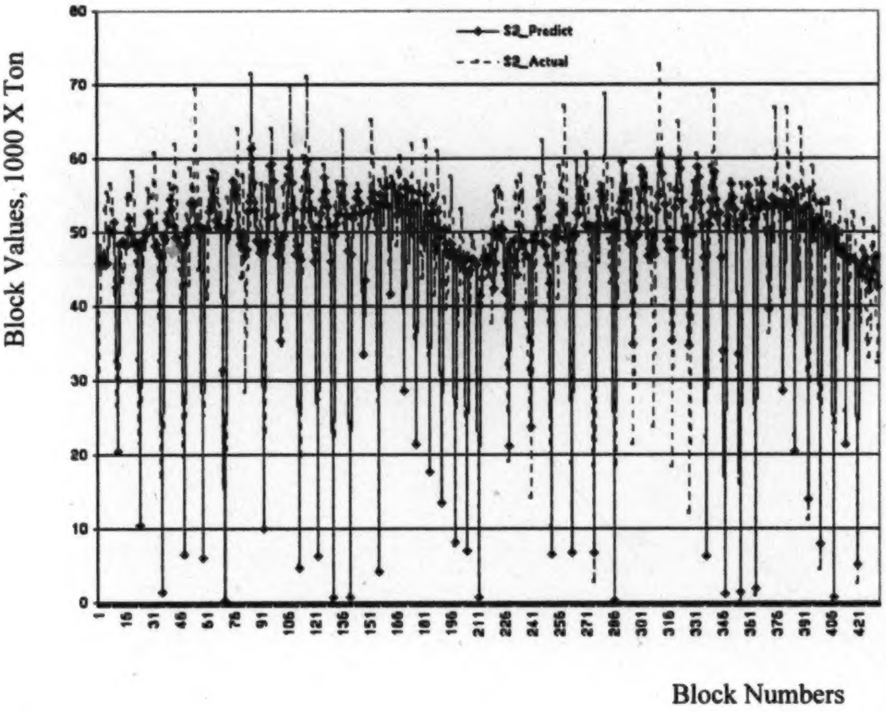


Fig. 7.22: Desired and Predicted Results from GRNN – Coal Seam 2

The best neural network is a generalized regression neural network. Figures 7.21 and 7.22 depict the best (king of the hill) models for the 2-seam coal deposit. The deviations in both seams 1 and 2 are small. The level of deviations (overestimates and underestimates) are not as pronounced as in the CATNN case (Figures 7.17 and 7.18). In seam 1 (Figure

7.21), the highest deviation is about 8,000 tons. The frequency of the deviations is also about one-third of those of the CATNN model. In a two-colour display where the differences are much more evident, the deviations estimated using a simple count is about 12%. The most interesting feature of this neural network model is that the model was able to accurately predict the depressed value at the washout.

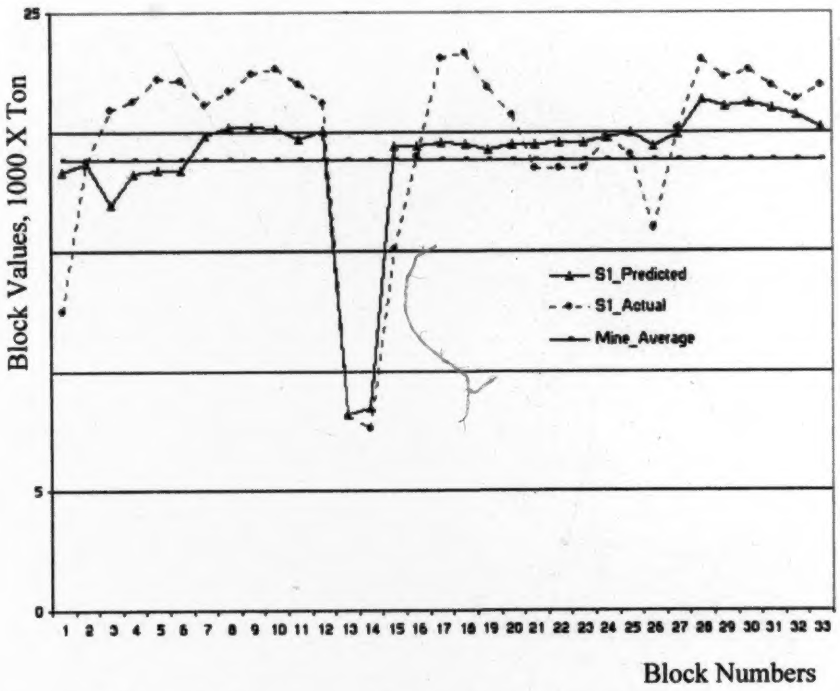


Figure 7.23: GRNN Experimental Results for Coal Block Model – Seam 1.

A similar picture is portrayed in the case of seam 2 (see Figure 7.22). The deviations are mostly underestimates. The model is also able to pick up the abnormally depressed block values in the middle of the block model.

The difference between the actual and the predicted values in the initial experiment is about 5,000 tons. The estimated/predicted value of about 17,000 tons is nearer to the average of 18,000 tons than the actual block value of 12,000 tons. The actual and predicted values are equivalent to each other in about 8 places in the graph (Figure 7.23). It is quite interesting to note that the predicted values are nearer the average value than the actual values and do not exhibit the erratic behaviour of the actual model.

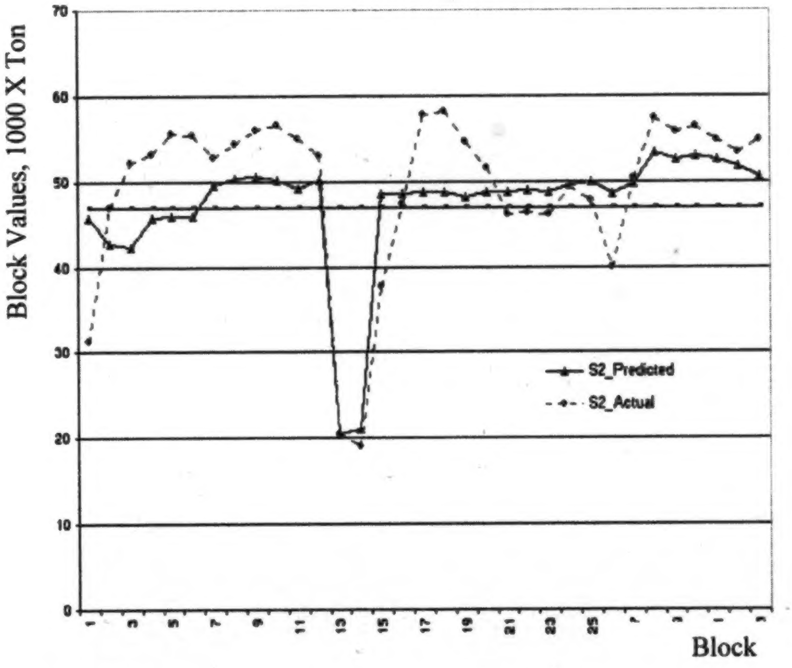


Figure 7.24: GRNN Experimental Results for Coal Block Model – Seam 2

The predicted model was able to follow the general trend of the actual model and was even able to accurately predict the trough in the actual values.

A similar trend is portrayed in the seam 2 case (see Figure 7.24). The difference in the initial estimates amount to about 13,000 tons. The two models cross each other in about 8 different places.

7.4 Significance of Prediction of Block Values Using the Neuro-Genetic Algorithm

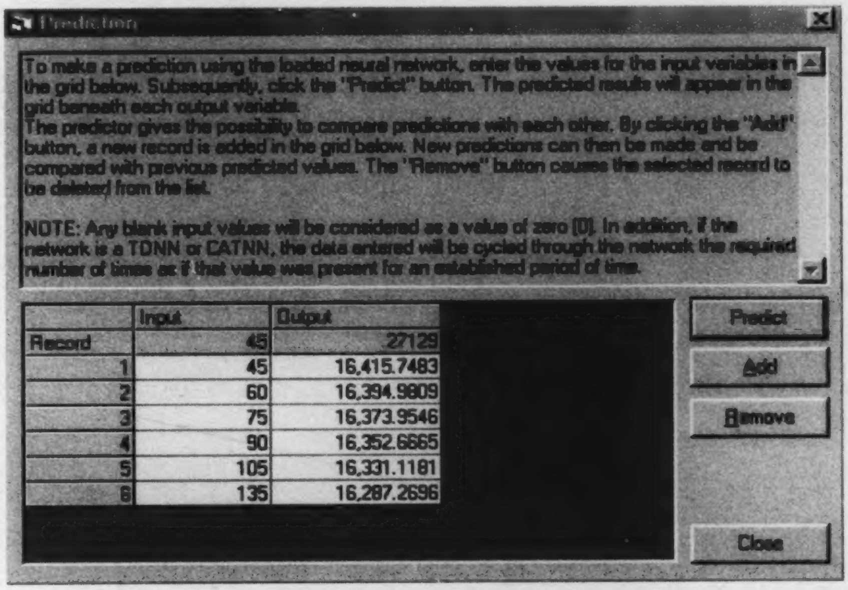
The backpropagation and continuous time neural networks failed in predicting block values accurately. Even though the NGO is a good and reliable estimation methodology, all the automatically generated predictors fell short of expectations. The reason for this is that the neurogenetic algorithm, like any other machine learning and or search algorithm can be considered as a black box or blind search when allowed to operate automatically. In this mode the NGO is given minimum information, obtains only scanty information from the environment as well and therefore outputs the wrong results. Even though the NGO has helped resolve one aspect of the neural network modelling problem - model building complexity, it cannot suffice as a black box. The main reason for the failure of BP and CATNN in both block models has to do with the stochastic nature of the underlying phenomena of the block data. Even though the backpropagation algorithm has its own problems, the absence of stochasticity in both the BP and CATNN could not be compensated for in any other way, leading to poor results.

However the generalized regression neural network emerged as the best modelling technique in both situations when a probabilistic approach was adopted. This is proof of the fact that a little knowledge goes a long way. According to the No Free Lunch (NFL) theorems of Wolpert and Macready (1996), it is entirely hopeless to use any algorithm to solve a problem without any prior knowledge. In fact a little knowledge is considered a quantum leap in modelling situations such as the above work. The GRNN model block values can be predicted from neuro-genetic optimizer with considerable accuracy.

The ability of a machine learning algorithm like the neuro-genetic algorithm to accurately learn and predict block values is a significant step in the automation of the mine planning and production processes. Prior knowledge and the accurate prediction of block values prior to mining will enable the automation of many mining production processes and can lead to considerable cost reduction and savings in plant chemicals. This breakthrough will actually serve as the basis of automation of open pit mine planning and production operations.

7.5 Results of Open Pit Optimization by Simulated Annealing and Neuro-genetic Algorithms

Due to the stochastic nature of the phenomena underlying the block grades, the output from the simulated annealing algorithm is input into the neurogenetic optimizer. It is also true that in an actual production operation, pit optimization is a dynamic process. The final pit may vary from what was conceptually envisaged in the initial stages of mining. However the entire pit design and optimization will have to be repeated every time there are changes. This is especially true in the Lerchs-Grossmann's algorithm. This is because it is a discrete optimization method and does not take the stochastic processes underlying the parameters into consideration. The process is thus tedious and repetitive and does not support a quick and effective mine production and managerial decision making in today's competitive environment.



To make a prediction using the loaded neural network, enter the values for the input variables in the grid below. Subsequently, click the "Predict" button. The predicted results will appear in the grid beneath each output variable.
The predictor gives the possibility to compare predictions with each other. By clicking the "Add" button, a new record is added in the grid below. New predictions can then be made and be compared with previous predicted values. The "Remove" button causes the selected record to be deleted from the list.

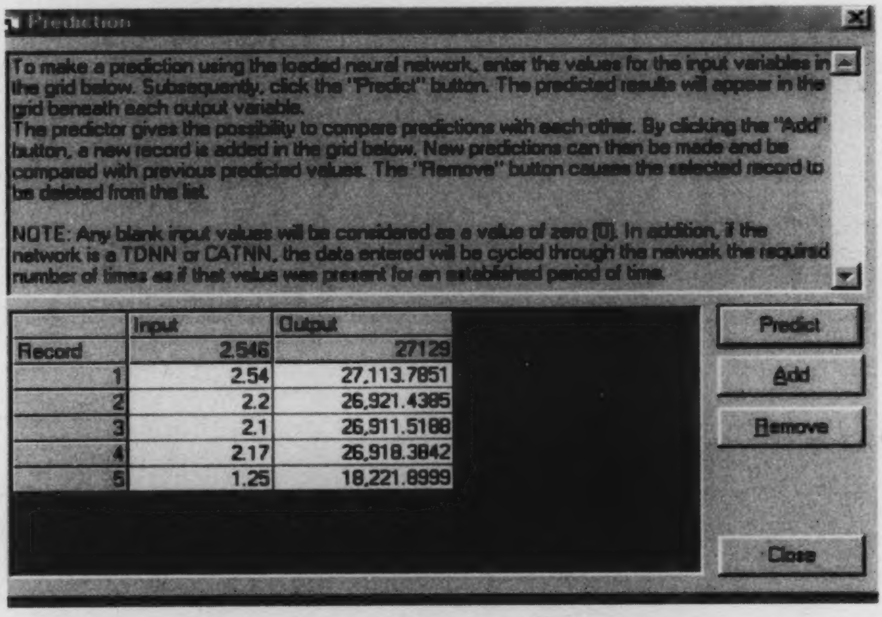
NOTE: Any blank input values will be considered as a value of zero (0). In addition, if the network is a TDNN or CATNN, the data entered will be cycled through the network the required number of times as if that value was present for an established period of time.

	Input	Output
Record	45	27129
1	45	16,415.7483
2	60	16,394.9809
3	75	16,373.9546
4	90	16,352.6665
5	105	16,331.1181
6	135	16,287.2696

Figure 7.25: Predictions from Neuro-Genetic Back-Propagation Network

In order to find an appropriate NGO to effectively build an optimized open pit model and predict pit values, all the neural network models were allowed to work during the optimization process. In this normal operating mode of the NGO, the back-propagation algorithm emerged as the best choice. When trial predictions were made from the resulting neural network, it became obvious that the output was not good enough in this particular case. Figure 7.25 is the results of predictions using the NGO. As depicted in

Figure 7.25, the values predicted from the BP neural network was short by about $10 million dollars. However using the same figures but disabling the back-propagation algorithm altogether, the resulting GRNN predictor could predict with much higher accuracy (see Figure 7.26). This figure depicts the learned relationship between the block grade and the optimized pit value.



To make a prediction using the loaded neural network, enter the values for the input variables in the grid below. Subsequently, click the "Predict" button. The predicted results will appear in the grid beneath each output variable.
The predictor gives the possibility to compare predictions with each other. By clicking the "Add" button, a new record is added in the grid below. New predictions can then be made and be compared with previous predicted values. The "Remove" button causes the selected record to be deleted from the list.

NOTE: Any blank input values will be considered as a value of zero (0). In addition, if the network is a TDNN or CATNN, the data entered will be cycled through the network the required number of times as if that value was present for an established period of time.

	Input	Output
Record	2.546	27129
1	2.54	27,113.7851
2	2.2	26,921.4385
3	2.1	26,911.5188
4	2.17	26,918.3842
5	1.25	18,221.8999

Figure 7.26: Predictions from Neuro-Genetic General Regression Neural Net

Once again the automatic generation of neural networks is fatally flawed and must not be encouraged (see section 7.4). Thus extra care must be taken during the modelling of neural network applications else the resulting networks/values may not be representative of the underlying phenomena.

In fact if the graph of this two methods are examined, it becomes quite obvious that the back-propagation algorithm could not learn the phenomena effectively at all (see Figures 7.27 and 7.28). The BP could only predict a straight line whereas the GRNN followed the trend of the desired results. This same thing happened when modelling the grade values. This may be also due to the nature of the back-propagation algorithm. It normally does a local optimization and can not effectively model hill-climbing and/or descent problems.

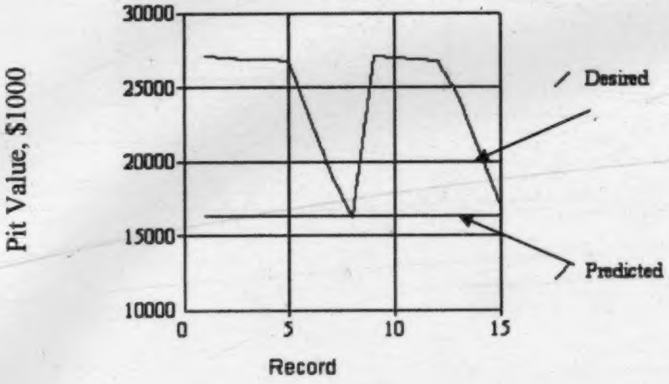


Figure 7.27: Graph of Desired Vs. Predicted Pit Values - BP

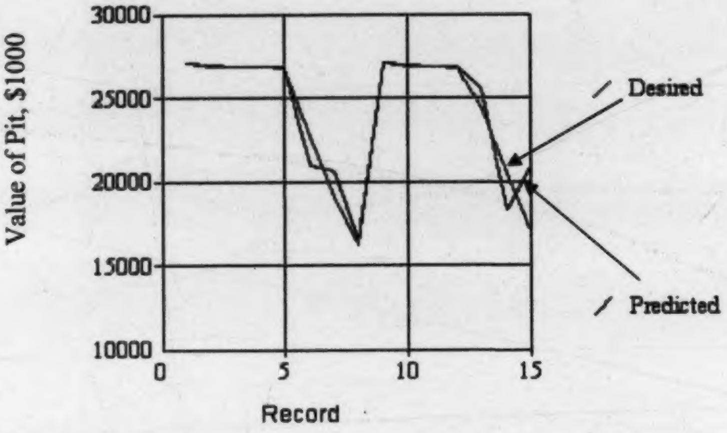


Figure 7.28: Graph of Desired Vs. Predicted Pit Values - GRNN

From Figure 7.29, a number of pits were generated by the simulated annealing algorithm. These pits range in value from over $15 million to over $27 million dollars. Finding the best/optimized pit to represent the ore body involves the use of the NGO as a predictor of pit values. When the optimum was reached the NGO values became asymptotic to an optimum value of about $27.5 million. The NGO converged and no optimum value significantly higher than this could be predicted with it.

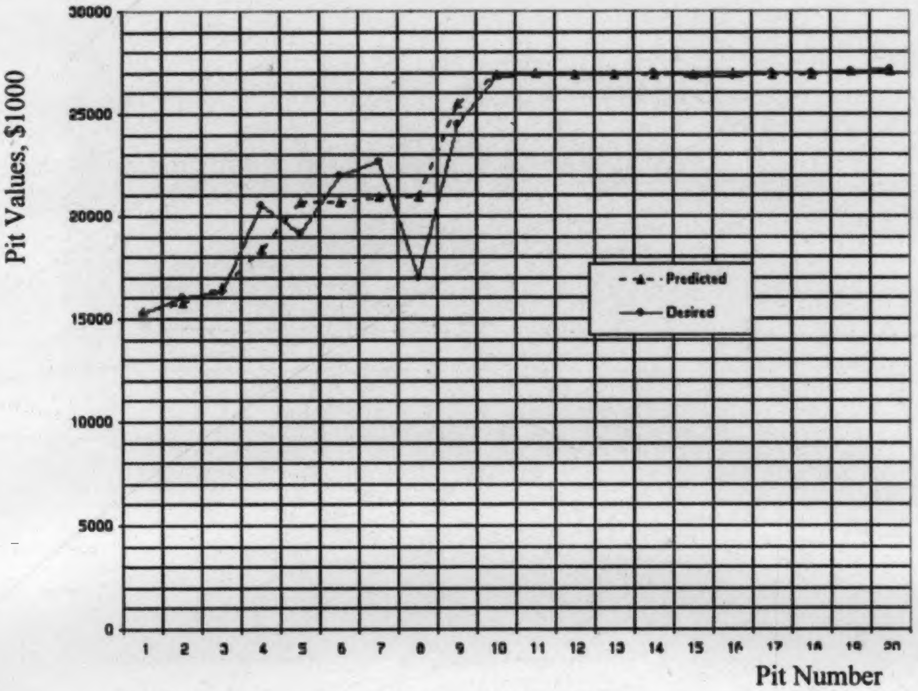


Figure 7.29: Open Pit Optimized Using the NGO

7.6 Sensitivity and Comparative Analysis of Results of Simulated Annealing – Neuro-Genetic and Lerchs-Grossmanns Algorithms

The results of further experimentation with the simulated annealing – neurogenetic optimizer (SA-NGO) and the Lerchs-Grossmann (LG) algorithms are plotted in Figure 7.30. In the initial stages the SA-NGO algorithm is lower than the L-G algorithm, but it rises faster than the LG algorithm. This behaviour can be explained by the fact that the LG algorithm is discrete whereas the SA-NGO algorithm is stochastic. Initially the LG algorithm performed better than the SA-NGO. The SA-NGO improved faster as block values were increased. The SA-NGO and the LG values were actually equivalent to each other at about 5 different places (see Figure 7.30). However the fast rise in the value of the SA-NGO levelled off around the optimum value of about $27.5 million. The value of the LG output however kept rising with rising block values. This can be a serious problem in pit design and may account for failure of about 25% of all new gold ventures. An important aspect of the results are the use of stochastic optimization which took into consideration the phenomena underlying the parameter employed in open pit mine design and optimization.

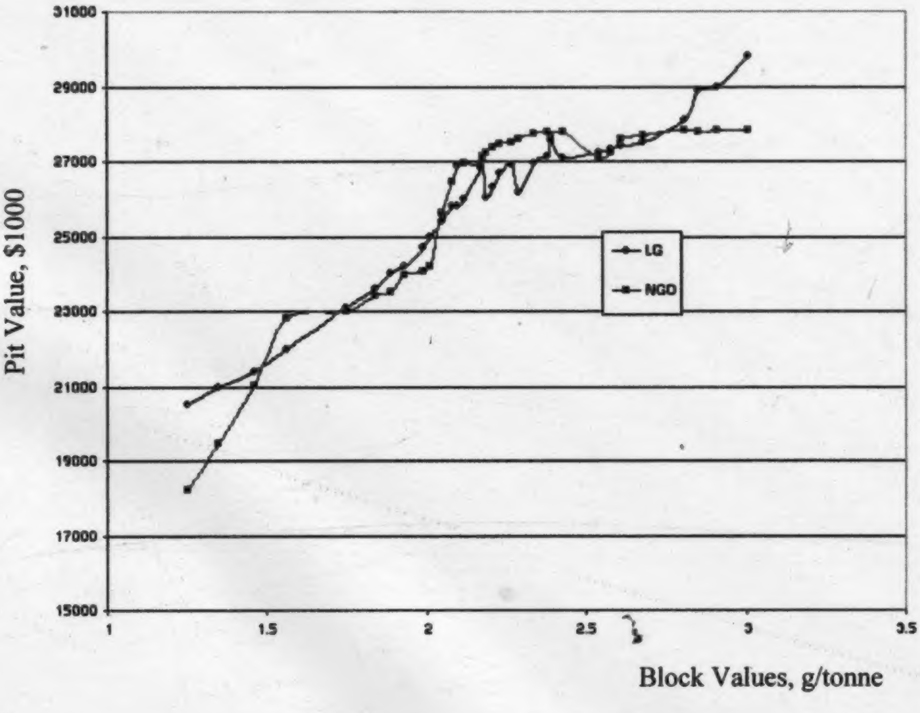


Figure 7.30: Graph of Lerchs-Grossmann and Neurogenetic Optimizer Pit Values

The results can also be very useful to mine operators. The various predictors like the block value and the optimized pit predictors can provide accurate answers for quick and reliable mine production decisions. The mine design and planning engineers may not have to re-invent the wheel any time any of the parameters change as in the case of the LG algorithm. This work also laid out a decent mine design and optimization procedures that are amenable to automation in an appropriate development environment.

The methodologies employed in this work are at the cutting-edge of today's information technology and it can be combined with data mining for effective mine production management decisions. The vital statistics for the comparative analysis are summarized in Table 7.1. The mean grade used was 2.20 g/tonne which lead to an average pit value of \$25,828.12 x 10^3 for the Lerchs-Grossmann and an average of \$25,839.67 x 10^3 for the SA-NGO algorithms respectively. The minimum grade employed in this study was 1.25 g/tonne which yielded pit values of \$20,500x10^3 and \$18221.9x10^3 for L-G and SA-NGO respectively. The stochastic characterization of the pit values is a beta distribution with the following statistics, namely chi-square test value of 77.963866, Kolmogorov test value of 0.274324 and Anderson-Darling test value of 1.640264.

The maximum grade of 3.01g/tonne led to pit values of 29,800x10^3 for L-G and 27521.1x10^3 for the SA-NGO (see Table 7.1). Thus the pit value of the LG algorithm kept increasing with increase in grade whereas the SA-NGO algorithm levelled off. This behaviour of the LG algorithm can be termed the augmented gold value effect. In this case the value of the pit increases with increasing grade even beyond the reasonable limits of the pit. The SA-NGO algorithm on the other hand does not exhibit this problem since it learns the features inherent in the pit values and then employs it to predict the value of the pit. Once the maximum pit value has been obtained the SA-NGO does not predict higher values even for higher grades.

Table 7.1: Summary Statistics of LG and SA-NGO

	Grade (g/tonne)	Lerchs-Grossmann ($'000)	SA-NGO ($'000)
Minimum	1.25	20500	18221.9
Maximum	3.01	29800	27521.1
Average	2.20	25828.12	25839.67
Standard Deviation	0.43	2293.94	2614.59

It is quite clear therefore that the use of L-G algorithm can lead to pit values higher than the actual value of the pit. This becomes extremely important in the case of a gold deposit in lieu of the high variability in gold grade values and the high failure rate of new gold projects. Thus the high failure rate of new gold projects can be partly attributed to this feature of the L-G algorithm. This apparent failure of the L-G algorithm to capture the true variability of the pit values can lead to escalated net income levels and higher than required investments which together spell doom for a mining project.

The graph in Figure 7.31 represents the relationship between the alpha values and confidence limits for both the LG and SA-NGO. In this case the confidence level is (100-α)%. The confidence limits for 99% confidence level are \$24,799.53x10^3 and \$26,856.71x10^3 for L-G and \$24,667.30x10^3 and \$27,012x10^3 for SA-NGO respectively.

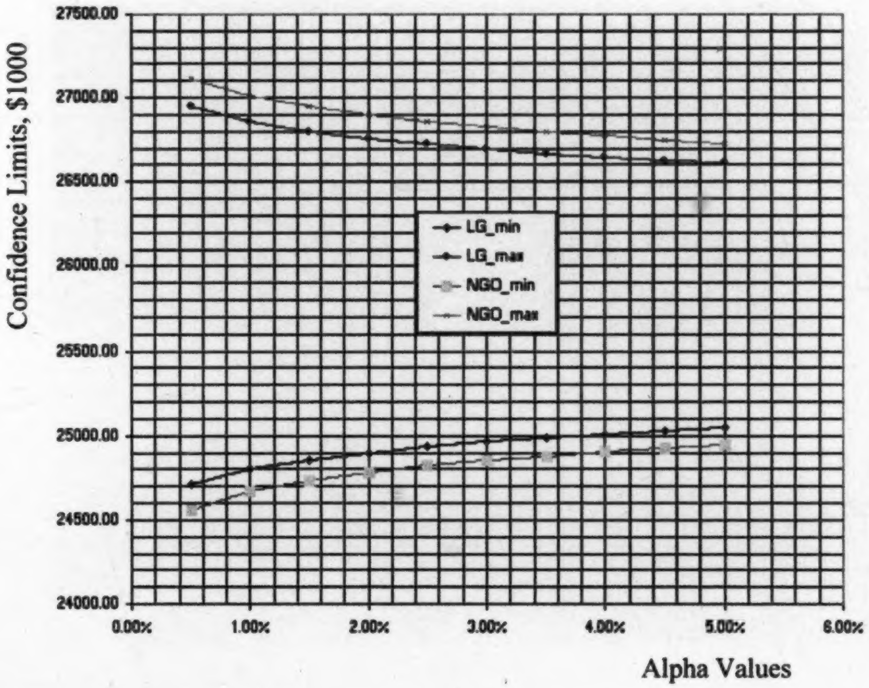


Figure 7.31: Graph of Alpha Values and Confidence Limits for LG and NGO Pits.

At a 95% confidence level, the confidence limits are \$25,045.46x10^3 and \$26,610.78x10^3 for L-G and \$24,947.61x10^3 and \$26,731.73x10^3 for SA-NGO respectively. The variability increases as the confidence level increases. The variability of the LG is lower than that of the SA-NGO. The LG algorithm is a discrete optimization method and can not therefore capture the true variability of the block grade values.

7.7 Stripping Ratio Results

The stripping ratio, which is defined as the quantity of waste to be removed in order to mine a unit quantity of ore is considered as one of the defining characteristics of open pit mines. The response of the value of the pit to a change in the stripping ratio for the Lerchs-Grossmann algorithm is depicted in Figure 7.32. Figure 7.33 represents the relationship between the pit value and the stripping ratio for the SA-NGO algorithm. It is quite evident from both Figures 7.32 and 7.33 that generally, the value of the pit decreases as the stripping ratio increases. An increase in stripping is tantamount to the mining of extra waste to gain access to the ore blocks. The specific behavior of the value of the ultimate pit in response to a change in the stripping ratio varies from one orebody to the other.

Figure 7.33 portrays the relationship between the pit values and the stripping ratio for the simulated annealing – neurogenetic algorithm. The downward nature of the general trend is evident once again.

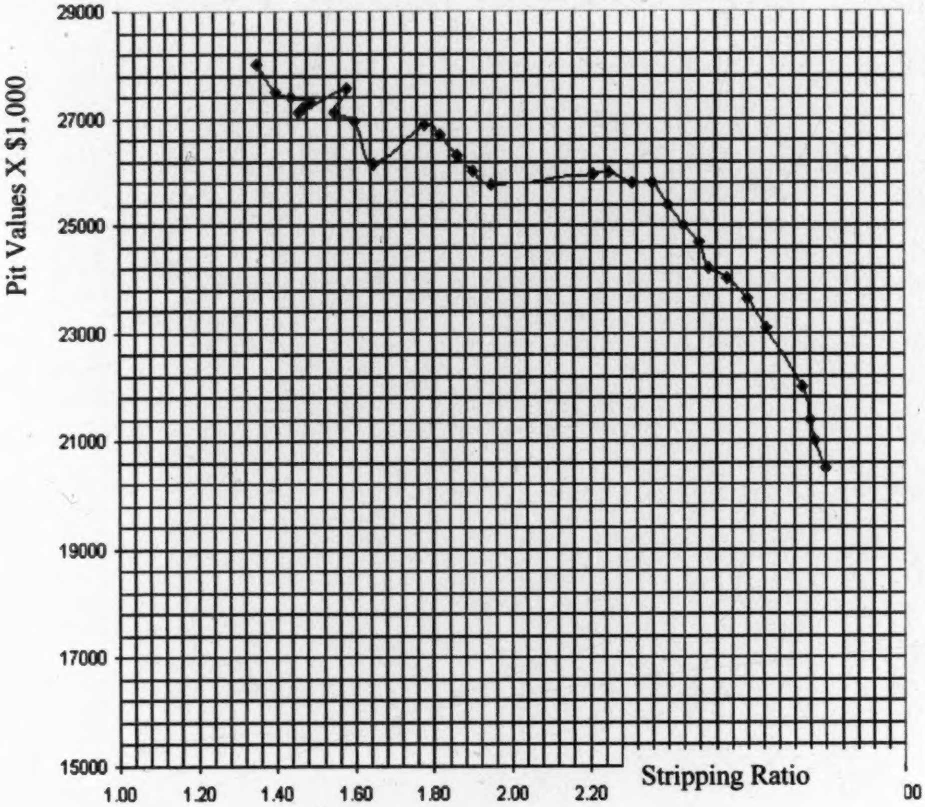


Figure 7.32: Graph of Pit Values versus Stripping Ratio for Lerchs-Grossmann Algorithm

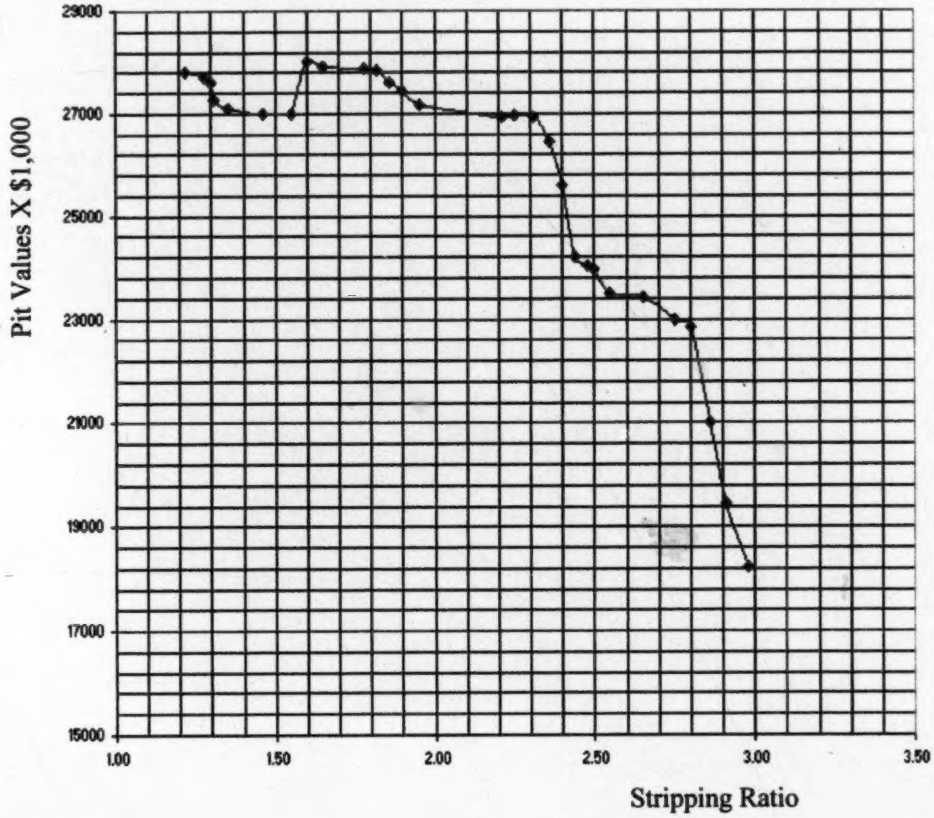


Figure 7.33: Graph of Pit Values versus Stripping Ratios for SA-NGO Algorithm

In Figure 7.32, an increase in the stripping ratio results in an decrease in the pit values. Though the general trend is downwards, the pit value increases momentarily when the stripping ratio increases. The downward trend is resumed again thereafter. The pit value increases just slightly again when the stripping ratio increases from 1.95 to 2.2. This is likely due to the topography.

The decrease in pit value is more pronounced and steeper beyond a stripping ratio of 2.3. The 2 pit value versus stripping ratio graphs have been plotted together in Figure 7.34.

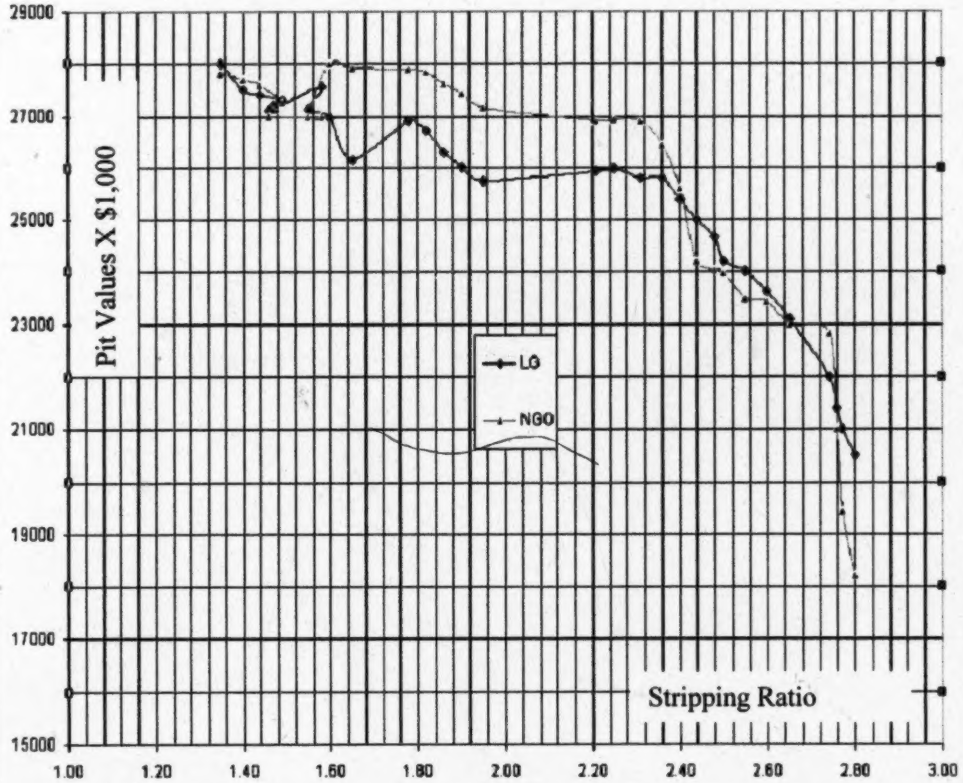
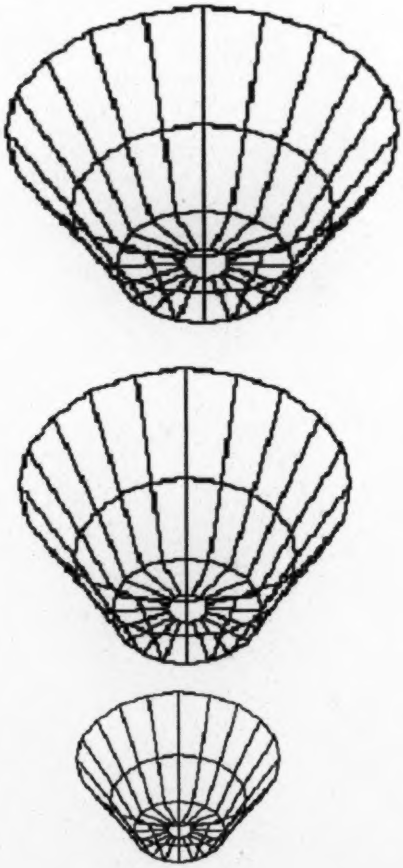


Figure 7.34: Graph of Pit Values versus Stripping Ratios for SA-NGO and L-G Algorithms

The L-G and SA-NGO values are the same at the low stripping ratios. The NGO pit value however increases and maintains a relatively higher value than the L-G.

7.8 Graphical Representation of An Optimized Open Pit Mine

Each of the long design together with the optimization procedures results in a single open pit. A graphical representation of three SA-NGO optimized pit developed using AutoCAD 2000 are shown in Figure 7.35.



Optimum value pit
with a total pit value
of $27.5211 million

Average value pit
with a total pit
value of
$25.83967 million

Minimum value pit
with a total pit value
of $18.2219 million

Figure 7.35: Three-Dimensional Representation of Three SA-NGO Pits

7.9 Conclusion

This chapter represents a discussion of the results of the design and optimization experiments performed using the algorithms and computer programs / software discussed in the earlier chapters. The computer software used during the experimentation are the Alnfit (Dendronic Decisions, 2000), NGO (Biocomp, 2000), GSLIB (Deustch, 1998) and AutoCAD (2000). In addition, two ANSI C programs written in MS Visual C++ 5.0 environment were also employed. Intelligent block models developed using machine learning algorithms like the ALN, MFNN with backpropagation learning and CATNN produced inaccurate results in both the gold and coal ore deposits. A stochastic approach, employing the GRNN was successful in both cases. The GRNN, MFNN and CATNN models were developed in a genetically-optimized learning environment. It became evident that the stochastic processes (natural phenomena) underlying the ore deposits must be taken into consideration in order to develop reliable models. A blind search or black box approach to intelligent computational modelling is bound to be futile, at best. The intelligent open pit model developed using a combination of simulated annealing and GRNN in a genetic environment outperformed the Lerchs-Grossmann's model in the prediction of the optimized pit. Both the intelligent block model and pit optimizer can be used to make just-in-time operational decisions as the models do not have to be re-run any time operational conditions change.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusions

The open pit mine design and optimization problem has been plaguing the surface mining industry for a long time. Various attempts have been made to address one form of this problem or the other. Most of these attempts were not aimed at only the optimization aspect of the problem, but also did not address the stochastic nature of the parameters involved in the design and optimization process. Though the neural network algorithm of Achireko and Frimpong (1996) addressed the stochastic nature of the input parameters, it is two-dimensional and therefore did not represent reality. Besides the short-comings of the back-propagation algorithm employed in this work, their work was also limited to the optimization aspects alone.

The main achievement of this work is the successful application of machine learning (adaptive logic network, neural network and genetic algorithms) to the complex processes underlying three-dimensional open pit mine design and optimization. A formalized procedure for geostatistical ore-body modeling and reserve evaluation using Geostatistical Software Library (GSLIB) has also been established. Machine learning was used to design a 3-D open pit mine. This has been accomplished in a gold as well as coal block model. The resulting neurogenetic mining block predictor can be effectively employed in automated mining/production systems. Simulated annealing and machine learning (neurogenetic) algorithms were then employed to optimize the designed open pit mine. Due to the complicated nature of the entire design and optimization process, it was broken down into various unit processes. The geological data for the gold mine was modelled using geostatistics and machine learning. The adaptive logic network and the neurogenetic algorithms were employed to build a block model predictor. Various machine learning algorithms were applied to the block model. The adaptive logic network employed 312 linear forms to fit the gold block model. The model error was high and the resulting ALN model could not predict the block values to an acceptable level of accuracy. This is because the ALN is sensitive to outliers and can not be effectively used to model data with sporadic changes. Predictions from the model developed using the feed-forward neural network with a back-propagation (genetic) learning algorithm were also inaccurate. There were wrong 50% of the time. A new intelligent block model was developed using the genetically configured generalized regression neural network (GRNN) algorithm. The architecture of this particular GRNN consisted of input and output layers, which are separated by 2 hidden layers. The first hidden layer consisted of 213 neurons and the second layer, 2 neurons. The correlation coefficient between the predicted and the actual values was 92%. Thus the GRNN neurogenetic block predictor was able to predict grade values at a considerably high accuracy and can be effectively used in an open pit operation to aid in grade control and other mining decisions. The GRNN neurogenetic predictor was therefore declared an acceptable block predictor. In the case of the coal model, two algorithms – the continuous adaptive time neural network and the generalized regression neural network models were employed. The highest level of underestimation in seam 1 obtained using the CATNN optimizer was 10,000 tons of coal. The highest level of overestimation in the same seam was 8,000 tons of coal. The average quantity of coal in seam 1 was 18,725 tons. The underestimation in the case of seam 2 amounted to about 18,000 tons of coal. The average quantity of coal in seam 2 is 46,981 tons. The predictions of the block values in both seams 1 and 2 using CATNN model could not follow the actual trend portrayed by the originating blocks. The major discrepancies between the actual and the predicted models in both seams occurred in the middle section of the deposit. Part of the coal seam has been washed away in the middle section but the CATNN model could not learn that. The best algorithm for both coal seams was once again the generalized regression neural network model. The estimated deviation using a simple count amounted to about 12% of the blocks in seam 1. The model was able to predict the depressed values at the position of the washout. These features were also true for the second seam.

The optimized 3-D open pit values for the L-G and SA-NGO algorithms are $29.8 million and $27.5211 million respectively. Even though the SA-NGO algorithm could learn the open pit values and effectively optimize the pit, the LG algorithm kept increasing in value with increase in block value. This could be misleading in a design and optimization environment and may lead to pit values higher than the actual value of that particular deposit. This work represents the first successful attempt at developing an intelligent 3-D block model and an open pit optimizer. In both the block model and open pit optimizer, the resulting intelligent predictors can effectively be used to predict block or pit values in an operational environment.

It evident from this study that a black box approach to the solution of a complex problem may not work. The solution methodology employed should take into consideration the phenomena underlying the various processes within the system under consideration. Stochastic processes must be modelled using stochastic neural networks. Even though machine learning algorithms like neural networks, adaptive logic network and genetic algorithms are very powerful tools, they must be used with care. Proper consideration must be given to the underlying phenomena. In this work a new modelling paradigm was successfully employed. Under this paradigm, a complex system / problem can be solved by breaking it down into units of smaller systems / problems. A recipe of appropriate solution methodologies can then be employed in solving the complex system or problem. The development of real-time intelligent 3-D block models for gold ore and coal is also significant. Block models developed using geostatistics alone can not be used off-line to predict block values in an advancing pit. The entire process of changing the parameters and generating the blocks again to make effective operational decisions can be tedious. This research has also resulted in an innovative design and optimization process. This new design methodology takes into account the stochastic (variability) nature of the parameters in the design process.

8.2 Recommendations

Even though the methods developed in this thesis has provided a very good model for open pit mine design and optimization, some specific differences may occur when applied to other mining ventures. There should be extensive studies to expand its usefulness to the mineral industry. Some of the recommendations for further research therefore are;

1. Studies aimed at incorporating some other modelling methods, neuro-genetic algorithms into orebody modeling and evaluation. Gaussian statistics may not be applicable in all cases.
2. Product, metal prices have not been effectively modelled and the use of neuro-genetic algorithms can be of tremendous benefit.
3. Aside from the methods described in this work, expert systems can be developed through the addition of knowledge, data-mining in appropriate environment for effective mine production decision making.
4. Appropriate procedures can be developed from this work to prevent software problems in the mining industry.

LIST OF REFERENCES

1. Achireko P. K., 1998 " Application of Modified Conditional Simulation and Artificial Neural Networks to Open Pit Optimization", Doctoral Thesis, Dalhousie University of Polytechnic, 179 pages.

2. Achireko P. K. and S. Frimpong, 1996, "Open Pit Optimization using Artificial Neural Networks on Conditionally Simulated Blocks"; Proc. 26th APCOM; Pittsburgh ; pp. 144-137.

3. Anon, 1996, "Giving Value to Mining Projects", Mining Magazine, August, 1996, p. 105.

4. Armstrong, W. W. and Bochmann, G. V., 1974, "Properties of Boolean Functions with a Tree Decomposition", BIT 13, pp. 1-13.

5. Armstrong, W.W., et. al. 1995, "Feasibility of Using Adaptive Logic Networks to Predict Compressor Unit Failure", Proceedings, Battelle Pacific Northwest Laboratories Workshop on Environmental and Energy Applications of Neural Networks, March 30-31.

6. Armstrong, W.W., et. al. 1995, "Adaptive Logic Networks in Rehabilitation of Persons with Incomplete Spinal Cord Injury", Proceedings, Battelle Pacific Northwest Laboratories Workshop on Environmental and Energy Applications of Neural Networks, March 30-31.

7. Atkinson, T. and Walton, G., 1983, "Design and Layout of Haul Roads for Surface Mines", Proceedings Symposium on Surface Mining and Quarrying, IMM, Bristol, UK, Oct. 4-6, pp. 84-91.

8. Axelson, A. H., 1964, "A Practical Approach to Computer Utilization in Mine Planning", Quarterly of the Colorado School of Mines, pp. 593- 622.

9. AutoCAD, Autodesk Inc.

10. Baron, Ken, 1997, "Class Notes – Rock Mechanics", University of Alberta, Edmonton.

11. Beale R. and T. Jackson, 1990, Neural Computing; IOP Publishing Ltd, New York, NY.

12. BioCOMP Systems Inc, 2000, Redmond, Washington.

13. Bishop, A.W., 1955. "The Use of the Slip Circle in the Stability Analysis of Slopes", Geotechnique, Vol. 5, No. 1, pp. 7-17.

14. Budd, Timothy A., 1997, "An Introduction to Object-Oriented Programming" 2nd Edition, 452pp.

15. Call, R.D. and Savely, J.P., 1990 " Open Pit Rock Mechanics" in Surface Mining, Society of Mining Engineers, Kennedy, B.A., Editor, Littleton, Colorado, pp. 860-882.

16. CANMET, 1976, Pit Slope Manual, Canada Centre for Mineral and Energy Technology (CANMET), Ottawa, Ontario, 10 Chapters with Supplements.

17. Carlson, S., 1997, "Algorithm of the Gods", Scientific American, March 1997, pp. 121-123.

18. Carlson, T.R, Erickson, J.P., O'Brian, D.T.., et al., 1966, "Computer Techniques in Mine Planning", Mining Engineering; Vol. 18, No.5; pp.53-56.

19. Cavender, B. "Determination of the Optimum Lifetime of A Mining Project Using the Discounted Cash Flow and Option Pricing Techniques" Mining Engineering, October, 1995, Society for Mining, Metallurgical, and Exploration, Inc., Littleton, Colorado, pp. 1262-1268.

20. Chadwick, J., 1996, "Haul Roads", Mining Magazine, July, pp. 30-33.

21. Choquet, P. and Tanon, D. D. B., 1985. "Nomograms for the Assessment of Toppling Failure in Rock Slopes", 26th U.S. Symposium on Rock Mechanics, South Dakota School of Mines and Technology, June 26-28, 1985, Rapid City, South Dakota, A. A. Balkema Press, Rotterdam, Netherlands, Vol. 1, pp. 19-30.

22. Clark, I., 1986, "The Art of Cross Validation in Geostatistical Applications", Proceeding of the 19th International Symposium on the Application of Computers and Operations.

23. Coleou, T. , 1989; "Technical Parameterization of reserves for Open Pit Design and Mine Planning"; 21st APCOM Proceedings; Littleton, Colorado; pp 485-494.

24. Coates, D.F. 1977, Pit Slope Design Manual, Chapter 5-Design, CANMET Report 77-5, 125 pages.

25. Collins, J.-L. "Evaluating Pit Designs Through DATAMINE-GUIDE Mining Software", International Journal of Surface Mining and Reclamation, (c) A. A. Balkema, Rotterdam, Netherlands, Volume 9, Number-- pp. 121-124.

26. Collins, J.-L., 1994, Converting An Optimum Pit Based on Block Modeling into a Practical Pit Based in String Modeling", International Journal of Surface Mining and Reclamation, (c) A. A. Balkema, Rotterdam, Netherlands, Volume 7, Number-- pp. 167-169.

27. Crawford, J. T., and Davey, R. K., 1979, "Case Study In Open-Pit Limit Analysis", Computer Methods for The 80s, Soceity of Mining Engineers of AIME, New York, pp. 310-319.

28. Cressie, N. ,1990, "The Origins of Kriging", Mathematical Geology, Vol. 22, No. 3, pp. 239-252.

29. Cundall, P.A 1987, "Distinct Element Models of Rock and Soil Structure". Analytical and Computational Methods in Engineering Rock Mechanics (E. T. Brown, Edited), George Allen and Unwin, London, pp. 129-162.

30. Dagdelen, K and Francois-Bongarcon, D., 1982, "Towards the Complete Double - Parameterization of Recovered Reserves in Open Pit Mining", Proceedings of the 17th International Symposium on the Application of Computer Methods in the Mineral Industry, Society for Mining, Metallurgy and Exploration, Littleton, Colorado.

31. Carvers, D. S., 1981, "Simple Methods to Analyse Buckling of Rock Slopes", Rock Mechanics, Vol. 14, No. 2, pp. 87-104.

32. Dagdelen, K., 1992, "Cutoff Grade Optimization", Proceeding's of the 23rd International Symposium of the Application of Computers and Operations Research in the Mineral Industry, University of Arizona, Tucson, Arizona, April, 1992. pp. 158-165.

33. Dendronic Decisions Ltd, 2000, Edmonton, Alberta.

34. Deutsch, 1998. Class Notes for MINE 750. University of Alberta, Edmonton, Alberta.

35. Deutsch, C. V. and Journel, A. G., 1998, GSLIB: Geostatistical Software Library and User's Guide, 2nd Edition, Oxford University Press, New York, New York, 369 pages.

36. Dowd, P.A. 1994, "The Optimal Design of Quarries"; Mineral Resource Evaluation II: Methods and Case Histories; Geological Society Special Publication No. 79, pp 141-155.

37. Dowd, P.A. and Onur A.H., 1992, "Optimizing Open pit Design and Sequencing"; Optimal Open-pit Design"; 23rd APCOM Proceedings; Littleton, Colorado; pp 411-422.

38. Dowd, P.A. and Onur. A.H., 1993, "Open Pit Optimization-Part 1: Optimal Open Pit Design" Transactions of Institute of Mining and Metallurgy (Section A: Mining Industry), May-August, 1993, pp. A95-A104.

39. Dowd, P.A. and Onur. A.H., 1993, "Open Pit Optimization-Part 2: Production Scheduling and Inclusion of Roadways" Transactions of Institute of Mining and Metallurgy (Section A: Mining Industry), May-August, 1993, pp. A105-A113.

40. Duzgun, H. S. B. and Pasamehmetoglu, A. G., 1995, "Plane Failure Analysis of Rock Slopes: A Reliability Approach", International Journal of Surface Mining and Reclamation, A. A. Balkema, Rotterdam, Netherlands, Volume 9, Number 6 pp. 1-6.

41. Ercelebi, S.G., and Yegulalp, T. M., 1993, "Reliability and Availability Analysis of Mining Systems", Transactions of Institute of Mining and Metallurgy (Section A: Mining Industry), 102, January-April, 1993, pp. A51-. (c) IMM London UK.

42. Erickson, J. D., 1968, "Long-Range Open Pit Planning", Mining Engineering, April, 1968, pp. 75-78.

43. Farifield, J.D. and Leigh, R. R., 1969, "A Computer Program For The Design of Open Pits", Quarterly of the Colorado School of Mines, pp. 329- 339.

44. Feknous, N., et. al., 1995, "Slope Stability of the Eastern Wall of Copper Mountain Open Pit Mines, Gaspe, Quebec", Third Canadian Conference on Computer Applications in the Mineral Industry, Montreal, Quebec, October 22-25, 1995, pp. 303-310.

45. Fensel, Dieter and Motta, Enrico, 1996, " Structured Development of Problem-Solving Methods", IEEE Transactions on Knowledge and Data Engineering, 1996.

46. Francois-Bongarcon, D. M. and A. Marechal, 1976, "A New Method Open Pit Design: Parameterization of the Final Pit Contour"; 14th APCOM Proceedings; New York, New York; pp 573 -583.

47. Francois-Bongarcon, D. M. and D. Guibal, 1983a, " Parameterization of Optimal Design of an Open Pit "; AIME Transaction Vol.274; pp1801-1805.

48. Frimpong, Samuel, 1991, "Profitability-Sensitivity and Risk Analysi of the Sabi Gold Project", M.Sc. Thesis, University of Zambia, Lusaka, Zambia.

49. Gomm, J. B., Page, G. F. and D. Williams, 1993, "Introduction to Neural Networks"; Application of Neural Networks to Modelling and Control ; pp 1-7.

50. Goodman, R. E., 1980, Introduction to Rock Mechanics, John Wiley and Sons, New York, New York, 478 pp.

51. Goodman, R. E. and Bray, J. W., 1976, "Toppling of Rock Slopes", In Proceedings, Speciality Conference on Rock Engineering for Foundations and Slopes, Boulder, Colorado, American Society of Civil Engineers, New York, Vol. 2, pp. 201-234.

52. Goovaerts, P., 1997, Geostatistics for Natural Resources Evaluation, Oxford University Press, N. Y, N. Y., 483 pages.

53. Green G., 1997, "Modelling Concept Design Evaluation", Artificial Intelligence for Engineering Design, Analysis and Manufacturing", © 1997 Cambridge University Press, Vol. 11, pp. 211-217.

54. Grossmann, I. F. and H. Lerchs, 1965, "Optimum Design of Open-Pit Mines"; Transactions, C.I.M., Volume LXVIII; Montreal; pp17-24.

55. Grotschel, M. and Lovaz, L., "Combinatorial Optimization: A Survey", DIMACS, Technical Report 93-29, Rutgers University, New Jersey, May 1993, 57 pages.

56. Hagan, M. T., Demuth, H. B. and Beale, M., 1995, Neural Network Design, PWS Publishing, Boston, MA, pages 1-1 to 19-21.

57. Hartman, H. L., 1987, Introductory Mining Engineering; John Wiley and Sons Inc, New York; p 149.

58. Haykin, S., 1994, Neural Network: A Comprehensive Foundation, IEEE Press; pp. 179-181.

59. Heckerman, D., 1996, "A Tutorial on Learning With Bayesian Networks", Microsoft Research, Redmond, WA, Revised November 1996, 57 pages.

60. Hertz, J., Krogh, A. and R. G. Palmer, 1991, Introduction to the Theory of Neural Computation, Adison-Wesly Publishers, MA., U.S.A.

61. Hittinger, M., 1978, "Numerical Analysis of Toppling Failures in Jointed Rock Masses", Ph.D. Thesis. University of California, Berkeley, 297 pp.

62. Hodge, R. A. L. and Freeze, R. A., 1977, "Groundwater Flow Systems and Slope Stability", Canadian Geotechnical Journal, Vol. 14, pp. 466-476.

63. Hoek, E. and Bray, J. W., 1981, Rock Slope Engineering, 3rd edition, Institute of Mining and Metallurgy, London, 402 pages.

64. Holmstrom, L, et. al., 1996, "Comparison of Neural and Statistical Classifiers – Theory and Practice", Research Report A13, Rolf Nevanlinna Institute, University of Helsinki, Helsinki, Finland.

65. Hopfield, J. J., 1982, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities"; Proceeding of National Academy of Science U.S.A., 79, pp 2554-2558.

66. Hopfield, J.J., 1984, "Neurons with Graded Response have Collective Computational Properties Like Those of Two-State Neurons"; Proceeding of National Academy of Science U.S.A., pp 3088-3092.

67. Hustulid, W., 1990, Fundamentals of Open Pit Mine Planning and Design Class Notes, Colorado School of Mines, Golden, CO.

68. Hustrulid, W. and M. Kuchta, 1995, Open Pit Mine Planning & Design, Volume I - Fundamentals; A.A.Balkema Publishers; Rotterdam, Netherlands; pp 413-442.

69. Intergraph Software, Intergraph Corporation, Hunstville, Alabama.

70. Issaks, E. H. and Srivastava, R. M., 1989, An Introduction to Applied Geostatistics, Oxford University Press, New York, 561 pages.

71. Janbu, N., 1954, "Application of Composite Slip Circles to Stability Analysis". Proceedings of European Conference on Stability of Earth Slopes, Stockholm, Statens Reproducktionsanstalt, Vol. 3, pp. 43-49.

72. Johnson, T.B. and Barnes R. J., 1988, "Application of Maximal Flow Algorithm To Ultimate Pit Design"; Engineering Design: Better Results Through Operations Research Methods; North Holland Publication Company; pp 518-531.

73. Johnson, T.B. and Mickel, D. G., 1970, "Optimum Design of an Open Pit: An Application in Uranium"; 17th APCOM Proceedings, CIM Special Volume 12; Montreal, Canada; pp 331-338.

74. Johnson, T.B. and W. R. Sharp, 1971, "A Three-dimensional Dynamic Programming Method for Optimal Open Pit Design"; Rep. Invest. U.S. Bureau of Mines, No. 7553; p25.

75. Johnson, T.B., 1973, "A Comparative Study of Methods for Determining Ultimate Open Pit Mining Limits"; 11th APCOM Proceedings; Tucson, Arizona: University of Arizona; pp 129-138.

76. Johnson. T. B. and Sharp, W. R., 1971, "A Three-Dimensional Dynamic Programming Method For Optimal Ultimate Open Pit Design", US Bureau of Mines Report on Investigations, No. 7553.

77. Journel, A. and C. H. Huijbregts, 1978, Mining Geostatistics, Academic Press, London.

78. Keaton, J. R. and Beckwith, G. H., 1996, "Important Considerations in Slope Design", Chapter 16, Landslides: Investigation and Mitigation, Special Report 247, Transportation Research Board, National Research Council, pp. 429-438.

79. Kennedy B. A.(Editor), 1990; Surface Mining, 2nd Edition, SME, Littleton; CO; pp. 460-485.

80. Kim, Y.C., 1978, "Ultimate Pit Limit Design Methodologies Using Computer Models - The State of The Art", Mining Engineering, Society of Mining Engineers, pp. 1454-1457.

81. Koborov, S., 1974, "Method for Determining Optimal Open Pit Limits; Rapport Report EP 14R-4, Ecole Polytechnique de Montreal.

82. Koenigsberg, E., "The Optimum Contours of An Open Pit Mine: An Application of Dynamic Programming", 17th APCOM Symposium, pp. 274-286.

83. Kohonen, T., 1988, Self-Organization and Associative Memory; 2nd Edition, Springer-Verlag, Berlin and Heidelberg.

84. Kosko, B., 1994, "Fuzzy Systems as Universal Approximators", IEEE Transactions on Computers, Vol. 43 (11), pp. 1329-1333.

85. Kreps, D.M, 1990, A Course in Microeconomic Theory, Princeton University Press, Princeton, New Jersey.

86. Krige, D., 1951, "A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand", M. S. in Engineering thesis, University of Witwatersrand.

87. Lane, W., 1997, "Bankable Feasibility Studies: Five Bankers Point the Way", Engineering and Mining Journal, November, pp. 16SS-16CCC.

88. Lemieux M., 1968, "A Computerized Three Dimensional Optimum Open Pit Design System"; CIM Annual Meeting, Montreal, Canada.

89. Lemieux, M., 1976, "A Different Method of Modelling A Mineral Deposit For A Three- Dimensional Open Pit Computer Design Application", Proceedings of the 14th APCOM Symposium, Pennsylvania State University, Chapter 44, pp. 557-571.

90. Lemieux, M., 1979, "Moving Cone Optimizing Algorithm", Computer Methods for The 80s, Soceity of Mining Engineers of AIME, New York, pp. 329-345.

91. Lerchs, H. and Grossmann, I.F., 1965, "Optimum Design of Open Pit Mines", The Canadian Mining and Metallurgical Bulletin, January, Montreal, pp. 47-54.

92. Lorig, L.J, et al., 1991, "Slope Stability Analysis of Jointed Rock Mass Using the Distinct Element Method" Transportation Research Record 1330, TRB, National Research Council, Washington, D.C, pp. 1-9.

93. Luppens, J.A., et. al., 1996, " The Use of PC Software for Reserve Evaluation and Mine Planning", Preprint Number 96-114, Society for Mining, Metallurgical, and Exploration, Inc., Littleton, Colorado.

94. Marino, J. M. and Slama, J. P., 1973, "Ore Reserve Evaluation and Open Pit Planning"; Proceedings of 10th APCOM; South African Institute of Mining and and Metallurgy, Johannesburg; South Africa, pp. 139-144.

95. Mastoris, J., and Topuz, E., 1995, "Modeling, Optimization and Sensitivity Analysis of the Final Pit Limits for A Lignite Deposit", Mining Engineering, November, 1995, Society for Mining, Metallurgical, and Exploration, Inc., Littleton, Colorado, pp. 1027-1032.

96. Matheron, G., 1970, "Random functions, and their applications in Geology", in Geostatistics – A Colloqium (D. Merriam, ed.), Plenum Press, New York, pp. 79-87.

97. Matheron, G., 1973, "The Intrinsic Random Functions and their Applications", Advanced Application of Probability., Vol. 5, pp. 439-468.

98. Matthews, W. H. and McTaggert, K. C., 1969, "The Hope Land Slide, British Columbia", Proceedings of Geology Association of Canada, Vol. 20, No. 65-B, pp.65-76.

99. McCulloch, W. and Pitts, W., 1943, "A Logical Calculus of Ideas Imminent in Nervous Activity", Bulletin of Mathematical Biophysics, Vol. 5, pp. 115-133.

100. McMahon, B. K., 1975, "Probability of Failure and Expected Volume of Failure in High Slopes" Proceedings of the 2nd Australia-New Zealand Conference of Geomechanics, Brisbane, pp. 308-317.

101. Metropolis, N. et. al., 1953. "Equation of State Calculation by Fast Computing Machines", Journal of Chemical Physics, 21 (6), pp. 1087-1092.

102. Meyer M., 1969, "Applying Linear Programming to the Design of Ultimate Pit Limits"; Management Science, Volume 16, No. 2; pp. B121-35.

103. Meyer, M., 1969, "Applying Linear Programming to the Design of Ultimate Pit Limits", Management Science, Vol. 16, No. 2, October, 1969, pp. B-121 to B-135.

104. Minsky, M., 1961, "Steps Towards Artificial Intelligence", Proc. of the IRE, Vol. 49, pp.8-30.

105. Mitchell, T. M., 1997, Machine Learning, McGraw-Hill, 413 pages.

106. Morgenstern, N.R., and Price, V. E., 1965, "The Analysis of the Stability of General Slip Surfaces", Geotechnique, Vol. 15, No.1, pp 79-93.

107. Moss, A. S. E. and Steffen, O. K., 1978, "Geotechnology and Probability in Open-Pit Mine Planning", Proceedings 11th Commonwealth Mining and Metallurgical Congress, IMM, Hong Kong, pp.543-553.

108. Murray-Smith, R. and Johansen, T. A. , 1997, Multiple Model Approaches to Modelling and Control, Taylor and Francis Publishers, London, U.K.

109. Neter J. et. al., 1990, Applied Statistical Models, 3rd edition, Richard D. Irwin Inc., Homewood, IL, pp. 465-471.

110. NeuroGENESYS, 2001, Honeywell Technology Center, Minneapolis, Minesotta.

111. Nilson, D.,1982, "Open-Pit or Underground Mining", SME Underground Mining Methods Handbook , SME of AIME; pp. 70-87.

112. Pana, M. and Davey, R. K., 1965,"The Simulation Approach to Pit Design; Proceedings of 5th APCOM; Arizona; pp. zz1-zz24.

113. Pana, M. T., and Davey, R. K., 1973, "Pit Planning and Design", Mining Engineering Handbook, Society of Mining Engineers of AIME, New York, pp. 17-10 to 17-19.

114. Paul, P. K. and De, P., 1989, "Matching of Shovel-Dumper at Production Planning Stage - a Mathematical Approach", Journal of Mines, Metals and Fuels, April, 1989, pp. 143-146.

115. Philips, D. A., 1972, "Optimum Design of an Open Pit"; Proc. 10th APCOM; SAIMMM, Johannesburg; South Africa; pp 145-157.

116. Phillips, D. W., 1989, Neural Computing: Theory and Practice, © Van Nostrand Reinhold, N. Y., N. Y. 230 pages.

117. Picard, J-C. and Smith, B. T., "Optimal Rate of Return in Open Pit Mine Design", Unknown pp. 111-118.

118. Pindyck, R. S. and Rubinfeld, D. L. 1991, Econometric Models and Economic Forecasts; McGraw-Hill Inc., New York; pp. 73-100.

119. Piteau, D. R. and Martin, D. C., 1977, "Slope Stability Analysis Design Based on Probability Techniques at Cassar Mine, CIM Bulletin, Vol. 70, March, 1977, pp. 139-150.

120. Priest, S. D. and Brown, E. T., 1983, "Probabilistic Stability Analysis of Variable Pit Slopes", Transactions of IMM, Section A: Mining Industry, No. 92, pp. A1-A12.

121. Pro/Engineer, Parametric Technology Corporation, Waltham, MA.

122. Russell, S. and Norvig, P., 1995, Artificial Intelligence: A Modern Approach, Prentice-Hall, Upper Saddle River, New Jersey, 932 pages.

123. Sarma, S. K., 1979, "Stability Analysis of Embankments and Slopes", Journal of the Geotechnical Engineering Division, ASCE, Vol. 105, No. GT12, pp. 1511-1524.

124. Seymour, F., 1995, "Finding the Mining Sequence and Cutoff Grade Schedule that Maximizes Net Present Value", Preprint Number 95-69, Paper at Presented SME Annual Meeting, Denver, Colorado, March 6-9, 1995.

125. Seymour, F., 1995, "Pit Limit Parameterization From Modified 3D Lerchs-Grossmann Algorithm", Preprint Number 95-69, Paper Presented SME Annual Meeting, Denver, Colorado, March 6-9, 1995.

126. Shang, Yi, 1997, "Global Search Methods for Solving Nonlinear Optimization Problems", Doctoral Dissertation, University of Illinois at Urbana-Champaign.

127. Shenggui Z. and Starfield A. M., 1985, "Dynamic Programming with Colour Graphics Smoothing for Open Pit Design on a Personal Computer" ; International Journal of Mining Engineering, Vol.3; pp 27-34.

128. Simon, H. A., 1981, The Sciences of the Artificial, MIT Press, Cambridge, MA.

129. Singh, V. K., et. al., 1995, "Slope Design Based on Geotechnical Study and Numerical Modeling of A Deep Open Pit Mine in India", International Journal of Surface Mining and Reclamation, A. A. Balkema, Rotterdam, Netherlands, Volume 7, Number 6 pp. 105-111.

130. Singh, M. P., 1994, Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications, Lecture Notes in Artificial Intelligence, Volume 799.

131. Spinellis, D. D. and Papadopoulos, C. T., 2000, "A Simulated Annealing Approach for Buffer Allocation in Reliable Production Lines", Annals of Operations Research 93, pp. 373-384.

132. Spinellis, D. D. and Papadopoulos, C. T., 2000, "Stochastic Algorithms for Buffer Allocation in Reliable Production Lines", Mathematical Problems in Engineering, Vol. 5 pp. 441-458.

133. Supynuk, A. G. and Armstrong, W. W., 1992, "Adaptive Logic Networks and Robot Control", Proceedings of Vision Interface Conference '92, also called AI/VI/GI'92, Vancouver, B. C., May 11-15, 1992, pp. 181-186.

134. Sutherland, I. E., 1963, Sketchpad: The First Interactive Computer Graphics, Ph.D. Thesis MIT, Cambridge, MA.

135. Tolwinski, B. and R. Underwood, 1992, "An Algorithm to Estimate the Optimal Evolution Pit Mine"; 23rd APCOM Proceedings; Tucson, AZ; pp. 399-409.

136. Tolwinski, B., 1996, "An Application of L-Tops to Project Evaluation", Preprint Number 96-77 and Presented at SME Annual Meeting, Phoenix, Arizona, Society for Mining, Metallurgical, and Exploration, Inc., Littleton, Colorado.

137. Turgeon, L. et. al., 1995, "Stability Analysis of An Open Pit Wall Using A Limit Equilibrium Method", Third Canadian Conference on Computer Applications in the Mineral Industry, Montreal, Quebec, October 22-25, 1995, pp. 311-320.

138. Turner, K. A. and Schuster, R. L.,, 1996. Editors: "Landslides-Investigation and Mitigation", Special Report 247, Transportation Research Board. National Research Council.

139. Wang, Q. and H. Sevim, 1992, "Enhanced Production Planning in Open Pit Mining Through Intelligent Dynamic Search"; 23rd APCOM Proceedings; Tucson, AZ; pp. 461-471.

140. Wang, Q. and Sevim, H., 1995, "Alternative to Parameterization in Finding a Series of Maximum-Metal Pits for Production Planning", Mining Engineering, SME, Littleton, Colorado, February, pp. 178-182.

141. Wasserman, P. D., 1992, Neural Computing: Theory and Practice; Van Nostrand Reinhold, New York; pp. 53-54.

142. Whitley, D., 1995, "A Genetic Algorithm Tutorial", Statistics and Computing Vol. 4, pp. 65-85.

143. Whitley, D., 1993, "An Executable Model of A Single Genetic Algorithm", Foundations of Genetic Algorithms.

144. Wilke F. L., Muellar K. and E. A. Wright, 1984, "Ultimate Pit Limit and Production Scheduling Optimizations; 18th APCOM Proceedings; London: IMM; pp 29-38.

145. Williams, C. E., 1970, "Computerized Mine Planning"; AIME, Tucson, Arizona; pp 97-104.

146. Wright, E. A., 1987, "The Use of Dynamic Programming For Open Pit Mine Design: Some Practical Implications", Mining Science and Technology, January, pp. 97-104.

147. Wright, E. A., 1990, "Open Pit Mine Design Models, An Introduction with Fortran/77 Programs", Series in Mining Engineering, Volume 8 1990, Trans Tech Publications, Clausthal-Zellerfield, FRG.

148. Wright, E. A., et. al., "Recent Findings in Open Pit Optimization" International Journal of Surface Mining and Reclamation, A. A. Balkema, Rotterdam, Netherlands, Volume 7, Number-- pp. 155-159.

149. Zurada, J.M, 1992, Introduction to Artificial Neural Systems; West Publishing Company, St. Paul, MN.,U.S.A.

APPENDICES

APPENDIX 5.1: Adaptive Logic Network Objects

Some of the objects employed in the ALN modelling process are;

ALNCALLBACKINFO - used to specify a user-defined callback function to receive notifications during the ALN training and evaluation process,

ALNCONFIDENCE - this is an object used to evaluate the confidence interval of ALN errors,

ALNDATAINFO - this particular object is used specifies the input data during an ALN training or evaluation process,

ALNCONSTRAINT - this structure specifies the properties of an ALN variable,

LTUSTATS - this holds the regression statistics of an ALN,

LTUWEIGHTSTATS - this object holds the regression statistics of an LTU weight,

VARINFO - this object maps ALN variable inputs to locations within a data block specified in a separate ALNDATAINFO object or by user-defined callback with the ALN_VECTORINFO notification,

ALNAbort - this is used to abort operations and clean up allocated resources

ALNAddMultiLayer - this object enables conversion of an LTU node into a subtree,

ALNAddLTUs - this object is called when an LTU is to be added to an existing LTU node of the ALN,

ALNAddTreeString - this object is called when a subtree needs to be added to an existing LTU node of an ALN. The ALN as well as the parent LTU node should have already been created, ALNCalcConfidence - this object calculates the lower and upper bound of the (confidence interval) on the errors produced by the ALN pointed to by pALN a pointer to an ALN object),

ALNCalcRMSError - the root-mean-square of an ALN specified by pDataInfo is estimated by this object,

ALNConfidencePLimit - this object is used to calculate the tail area (*pdblPLimit), such that there is a probability (dblSignificance), usually between 0.05 or 0.01, of such a tail area meeting the confidence bounds specified in pConfidence,

ALNConfidenceTLimit - this object calculates the probability *pdblTLimit that the confidence interval specified in pConfidence (pointer to an ALNCONFIDENCE structure) covers at least a minimum fraction of the error distribution,

ALNConvertDtree - this object is used to create a DTREE from an ALN,

ALNCreateALN - an ALN is created using this object,

ALNDestroyALN - this object destroys the ALN created by ALNCreateALN,

ALNEval - a created ALN is evaluated on one or more data points by this object,

ALNinvert - this is used to compute the inverse of the original ALN,

ALNLTUAnalysis - this performs a statistical analysis of the ALN,

ALNLTUFreeAnalysis - this is used to free the analysis buffer allocated by the system by the previous call to ALNLTUAnalysis,

ALNLTUStats - this objects retrieves and displays the LTU statistics from the analysis buffer obtained from a previous call to ALNLTUAnalysis,

ALNLTUWeightStats - this objects retrieves and displays the LTU weights from the analysis buffer obtained from a previous call to ALNLTUAnalysis, ALNQuickEval - this function is used for a quick evaluation of an ALN from an input vector supplied user,

ALNRand - this object is called to retrieve the next value from a pseudo-random number generator,

ALNRandFloat - this object is called to retrieve the next value from a pseudo-random number generator and convert the value to floating point,

ALNRead - this reads an ALN saved by using the ALNWrite function,

ALNSetAbortProc - this object is used to chain the abort procedure for another library when the ALNAbort function is called,

ALNSetGrowable - this object is used to enable a subtree growable during training,

ALNSRand this is called to re-initialize the seed value of the ALN pseudo-random number generator,

ALNTrain - this object trains an ALN on one or more data point specified by pDataInfo,

ALNWrite - this object writes an ALN to a file

APPENDIX 6.1: Part of Rock Type 1 Data

Au	Values from	Tabu	Mine	Project	
RX	#NAME?				
RY	#NAME?				
Z	#NAME?				
Rtype	- rock				
Au	#NAME?				
	2992	2059.6	4664.82	1	0
	2987.75	2055.35	4649.18	1	0
	2983.59	2051.19	4633.83	1	0
	3068.93	2136.53	4942.32	1	6.342
	3065.16	2132.99	4938.07	1	3.02
	3061.99	2130.02	4934.5	1	2.416
	3060.48	2128.59	4932.79	1	5.134
	3058.33	2126.58	4930.37	1	3.624
	3056.68	2125.03	4928.5	1	1.51
	3055.87	2124.27	4927.59	1	1.51
	3052.08	2120.71	4923.29	1	0
	3046.29	2115.27	4916.72	1	1.54
	3042.88	2112.08	4912.85	1	0.604
	3040.74	2110.07	4910.42	1	0
	3034.47	2104.18	4903.27	1	3.684
	3028.77	2098.83	4896.74	1	1.812
	3026.77	2096.95	4894.46	1	2.718
	3024.5	2094.82	4891.85	1	0
	3022.89	2093.31	4890.01	1	0
	3018.91	2089.57	4885.43	1	0
	3014.72	2085.64	4880.61	1	3.02
	3011.72	2082.82	4877.16	1	0
	3006.67	2078.08	4871.32	1	0
	3000.05	2071.87	4863.67	1	0
	2996.26	2068.31	4859.26	1	0
	2993.86	2066.06	4856.48	1	2.416

APPENDIX 6.2: Part of Rock Type 2 Data

Au	Values from	Tabu 0	Mine Project		
5		0			
RX	#NAME?		0		
RY	#NAME?		0		
Z	#NAME?		0		
Rt	ype -	rock 0			
Au	#NAME?		0		
	2990.21	2062.64	4852.24	2	0
	2987.19	2059.8	4848.71	2	4.53
	2984.59	2057.35	4845.68	2	0
	2982.57	2055.46	4843.33	2	0
	2981.26	2054.23	4841.79	2	1.51
	2979.14	2052.24	4839.31	2	0
	2974.19	2047.59	4833.51	2	1.42
	2970.76	2044.37	4829.49	2	0
	2968.37	2042.13	4826.68	2	0
	2965.78	2039.7	4823.63	2	0
	2964.6	2038.59	4822.25	2	0
	2961.35	2035.54	4818.42	2	6.976
	2957.14	2031.59	4813.45	2	0
	2954	2028.64	4809.73	2	0
	2952.41	2027.15	4807.85	2	1.208
	2948.1	2023.11	4802.75	2	0
	2939.15	2014.7	4792.11	2	0
	3066.38	2134.19	4943.54	2	2.9
	3059.78	2127.99	4939 2	0	
	3055.35	2123.83	4935.94	2	0
	3052.38	2121.04	4933.88	2	0
	3050.99	2119.74	4932.93	2	0.604
	3044.9	2114.01	4928.7 2	0	
	3039.16	2108.62	4924.72	2	0
	3038.35	2107.86	4924.16	2	0
	3027.51	2097.67	4916.6 2	0	
	3015.54	2086.43	4908.22	2	0
	3013.79	2084.79	4907 2	0	
	3010.55	2081.75	4904.72	2	0
	3006.54	2077.98	4901.9 2	1.208	
	2999.81	2071.66	4897.16	2	0
	2986.21	2058.88	4887.53	2	0
	2976.51	2049.76	4880.63	2	0.604
	2973.15	2046.61	4878.24	2	1.208
	2971.17	2044.75	4876.83	2	0
	2970.28	2043.91	4876.19	2	0
	2943.44	2018.7	4856.93	2	0

2911.79	1988.97	4833.97	2	0
2899.04	1976.99	4824.66	2	0
2890.63	1969.1	4818.53	2	0
2888.82	1967.4	4817.21	2	0
2886.17	1964.9	4815.28	2	0
2882.67	1961.61	4812.72	2	0
2879.45	1958.59	4810.37	2	0.604
2873.9	1953.38	4806.33	2	0
2868.39	1948.2	802.3 2	0	
2866.26	1946.2	4800.75	2	0
2862.92	1943.06	4798.32	2	0.362
3074.91	2142.32	4949.75	2	0
3072.35	2139.84	4949.11	2	0
3067.67	2135.3	4947.95	2	3.474
3062.63	2130.42	4946.71	2	0
3058.61	2126.52	4945.73	2	0
3052.11	2120.22	4944.15	2	1.872
3038.8	2107.31	4940.96	2	0
3027.69	2096.55	4938.34	2	1.208
3022.83	2091.83	4937.2 2	1.51	
3018.09	2087.24	4936.1 2	0	
3014.15	2083.42	4935.19	2	0
3010.75	2080.12	4934.41	2	0
3008.04	2077.5	4933.79	2	0
3004.15	2073.72	4932.9 2	1.48	
3000.25	2069.94	4932.02	2	0
2997.95	2067.71	4931.5 2	0	
2993.34	2063.25	4930.46	2	0
2989.35	2059.38	4929.57	2	0
2987.05	2057.15	4929.06	2	0
2983.86	2054.06	4928.35	2	0
2980.22	2050.53	4927.55	2	0
2976.46	2046.88	4926.72	2	0
2972.64	2043.18	4925.89	2	0
2927.62	1999.54	4916.38	2	0
2883.59	1956.86	4907.64	2	0
2875.55	1949.06	4906.06	2	0
2868.17	1941.91	4904.62	2	0
3074.89	2142.25	4950.46	2	0
3070.99	2137.57	4951.85	2	1.208
3065.74	2131.25	4953.74	2	0
3061.76	2126.48	4955.17	2	0
3057.55	2121.42	4956.69	2	2.234
3047.46	2109.3	4960.34	2	0
3038.28	2098.26	4963.69	2	0
3031.97	2090.68	4965.99	2	0

APPENDIX 6.3: Part of Block Model of a Gold Mine

X	Y	Z	Block Value
22.5	4	12	-1.812
22.5	4	20	-1.812
22.5	12	4	-1.739
22.5	12	12	-1.739
22.5	12	20	-1.739
22.5	20	4	-1.840
22.5	20	12	-1.840
22.5	20	20	-1.840
22.5	28	4	-1.887
22.5	28	12	-1.887
22.5	28	20	-1.887
22.5	36	4	-1.931
22.5	36	12	-1.931
22.5	36	20	-1.931
.			
.			
127.5	52	4	6.870
127.5	52	12	6.870
127.5	52	20	6.870
127.5	60	4	6.870
127.5	60	12	6.870
127.5	60	20	6.870
127.5	68	4	6.870
127.5	68	12	6.870
127.5	68	20	6.870
.			
.			
172.5	4	4	6.769
172.5	4	12	6.769
172.5	4	20	6.769
172.5	12	4	8.603
172.5	12	12	8.603
172.5	12	20	8.603
172.5	20	4	8.603
172.5	20	12	8.603
172.5	20	20	8.603
172.5	28	4	8.603
172.5	28	12	8.603

APPENDIX 6.4: Part of Block Model of a Coal Mine

Block	Overburden	Interburden	Seam1	Seam2
2701	104.754	349.18	10.4776	26.194
2702	112.926	376.42	12.5048	31.262
2703	118.239	394.13	17.5784	43.946
2704	119.496	398.32	19.2304	48.076
2705	116.433	388.11	19.292	48.23
2706	115.749	385.83	18.0152	45.038
2707	83.418	278.06	15.1032	37.758
2708	99.828	32.76	18.7712	46.928
2709	103.692	345.64	20.8712	52.178
2710	110.805	369.35	21.2408	53.102
2711	114.303	381.01	22.1984	55.496
2712	113.832	379.44	22.1088	55.272
2713	97.272	324.24	21.1064	52.766
2714	90.633	302.11	21.7056	54.264
2715	90.897	302.99	22.3888	55.972
2716	81.759	272.53	22.6072	56.518
2717	78.312	261.04	22.0024	55.006
2718	82.119	273.73	21.2016	53.004
2719	82.107	273.69	17.7016	44.254
2720	93.981	313.27	19.796	49.49
2721	104.895	349.65	16.8168	42.042
2722	103.752	345.84	12.992	32.48
2723	99.441	331.47	12.8688	32.172
2724	64.098	213.66	8.1536	20.384
2801	65.724	219.08	7.644	19.11
2802	86.688	288.96	15.064	37.66
2803	86.793	289.31	18.9672	47.418
2804	87.306	291.02	18.8664	47.166
2805	87.42	291.4	19.5496	48.874
2806	87.663	292.21	17.3432	43.358
2807	81.711	272.37	15.8424	39.606
2808	100.548	335.16	19.2304	48.076
2809	110.76	369.2	21.896	54.74
2810	98.79	329.3	21.9016	54.754
2811	107.823	359.41	22.6576	56.644
2812	92.835	309.45	22.0808	55.202
2813	91.203	304.01	21.2184	53.046
2814	89.946	299.82	21.728	54.32
2815	87.675	292.25	23.0776	57.694
2816	80.523	268.41	23.2848	58.212
2817	78.873	262.91	21.8176	54.544
2818	80.649	268.83	20.6584	51.646
2819	80.529	268.43	18.4408	46.102

2820	79.632	265.44	18.4296	46.074
2821	77.28	257.6	15.8312	39.578
2822	78.312	261.04	13.132	32.83
2823	76.542	255.14	12.6	31.5
2824	22.908	76.36	4.1776	10.444
2901	53.313	177.71	5.712	14.28
2902	85.644	285.48	17.4104	43.526
2903	85.977	286.59	18.4968	46.242
2904	85.782	285.94	18.4632	46.158
2905	87.303	291.01	19.768	49.42
2906	88.212	294.04	19.0568	47.642
2907	83.772	279.24	15.9544	39.886
2908	87.639	292.13	20.1208	50.302
2909	97.305	324.35	22.9656	57.414
2910	92.361	307.87	22.2992	55.748
2911	92.916	309.72	22.568	56.42
2912	92.016	306.72	21.9016	54.754
2913	90.951	303.17	21.3136	53.284
2914	88.92	296.4	21.896	54.74
2915	81.564	271.88	24.9424	62.356
2916	78.984	263.28	24.2928	60.732
2917	78.93	263.1	21.5488	53.872
2918	79.179	263.93	20.3672	50.918
2919	78.792	262.64	18.6816	46.704
2920	77.358	257.86	17.1472	42.868
2921	74.655	248.85	18.256	45.64
2922	76.305	254.35	13.02	32.55
2923	70.608	235.36	11.8272	29.568
2924	2.118	7.06	0.532	1.33
3001	40.305	134.35	4.0824	10.206
3002	85.323	284.41	17.7352	44.338
3003	85.119	283.73	18.0432	45.108
3004	84.138	280.46	17.8696	44.674
3005	86.739	289.13	20.3	50.75
3006	88.71	295.7	21.0448	52.612
3007	85.023	283.41	16.3464	40.866
3008	88.107	293.69	21.0336	52.584
3009	89.874	299.58	23.492	58.73
3010	93.795	12.65	22.6352	56.588
3011	92.04	306.8	22.5064	56.266
3012	91.281	304.27	21.8176	54.544
3013	90.585	301.95	21.4088	53.522
3014	87.675	292.25	22.3104	55.776
3015	77.565	258.55	26.7568	66.892
3016	78.318	261.06	24.7072	61.768
3017	78.246	260.82	1.168	52.92

APPENDIX 6.5: D45Blk.c Program

```
/*  This program generates a summary report from  of the block data files   */

/*  Just remember to change the file names             */


#include <stdio.h>

#define FILENAME "d3blk45.txt"

int main()

{

  /*  Declare and initialize variables.  */

  int number_of_blocks=0, k;

  double x, y, z, p, sum=0, max, min;

  FILE *d3blk45;


  /*  Open file.  */

  d3blk45 = fopen(FILENAME,"r");


  /*  While not at the end of the file,  */

  /*  read and accumulate information.   */

  while ((fscanf(d3blk45,"%lf %lf %lf %lf",&x,&y, &z, &p)) == 4)

  {

    number_of_blocks++;

    if (number_of_blocks == 1)

                                        max = min = p;
```

```
 sum += p;
 if (p > max)
                        max = p;
 if (p < min)
                        min = p;
}

/* Print summary information. */
printf("Number of Block Values: %i \n",
                     number_of_blocks);
printf("Average Block:       %.2f \n",
                     sum/number_of_blocks);
printf("Total Value of all the Blocks:       %.2f \n",sum);
printf("Maximum Block Value:       %.2f \n",max);
printf("Minimum Block Value:       %.2f \n",min);

/* Close file and exit program. */
fclose(d3blk45);
return 0;
}
/*-----------------------------------------------*/
```

APPENDIX 6.6: Sim3D.c Program

```
/* This program generates an optimized block summary report from  of the block data
files. It is a modification of the Algorithm of the Gods (Carlson, 1997)  */
/* Just remember to change the file names            */


#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>


#define TEMP_FACTOR 0.9
#define YES  1
#define NO    0
#define REVERSE               1
#define TRANSPOSE             0
#define NUMLISTITEMS 200        /* NUMBER OF ITEMS ON THE LIST. CHANGE
                                THIS NUMBER TO SUIT APPLICATION.*/


#define FILENAME "d3blk45.txt"
```

```
#define  ALEN(a,b,c,d,e,f,g,h)   sqrt(((b)-(a))*((b)-(a))+((d)-(c))*((d)-(c))+((f)-(e))*((f)-
(e))+((h)-(g))*((h)-(g)))

struct ListElement
{
/* DEFINE YOUR DATA STRUCTURE HERE. WE'VE ASSUMED SOMETHING
LISTED IN CARTESIAN COORDINATES.  IF YOU'RE DATA IS NOT IN
THESE COORDINATES, YOU WILL NEED TO MODIFY THIS STRUCTURE
AND ALSO MAKE THE NECESSARY CHANGES IN THE ENERGY FUNCTION. */

        float x;
        float y;
        float z;
        float p;
};

int CHECKBEST = NO;
float BESTENERGY;
struct ListElement **bestList;

main()
{       struct ListElement **list;
        int runLimit, sucLimit, numListItems, i, numSuc;
int Anneal();
float temperature, energy;
struct ListElement **setup();
FILE *outFile, *energyFile;
void printList();

float GetEnergy(), ener;

numListItems = NUMLISTITEMS;

list = setup(&energy, &temperature, &runLimit, &sucLimit, numListItems);

outFile = fopen("InitList.dat","w");
printList(outFile,list, numListItems);
fclose(outFile);

energyFile = fopen("energy.dat", "w");
fprintf(energyFile,"Temp\tEnergy\tnumSuc\n");

for(i = 1; i < numListItems; i++)
{
        numSuc = Anneal(list, numListItems, runLimit, temperature, sucLimit,
                numListItems);
        ener = GetEnergy(list,numListItems);


        printf("%f\t%f\t%d\n",temperature,ener,numSuc);

        fprintf(energyFile,"%f\t%f\t%d\n",temperature,ener,numSuc);



                /* Start checking for the best solution?*/


        if(numSuc <= runLimit*0.01 && CHECKBEST == NO) {

                CHECKBEST = YES;

                BESTENERGY = ener;

        }


        if(numSuc == 0) break;


        temperature *= TEMP_FACTOR;

}


fclose(energyFile);


outFile = fopen("FinalList.dat","w");

printList(outFile,list, numListItems);

fclose(outFile);
```

```
    outFile = fopen("BestList.dat","w");
    printList(outFile,bestList,numListItems);
    fclose(outFile);

    outFile = fopen("lowestEnergy.dat","w");
    fprintf(outFile,"Lowest Energy = %f\n",BESTENERGY); /* Store energy. */
    fclose(outFile);

}

void printList(outFile,list, numListItems)
FILE *outFile;
struct ListElement **list;
int numListItems;
{
    int i;

    for(i=0; i< numListItems; i++)
        fprintf(outFile,"%f\t%f\n",list[i]->x,list[i]->y,list[i]->z,list[i]->p);

}
```

```
struct ListElement **setup(energy, temperature, runLimit, sucLimit, numListItems)
int *runLimit, *sucLimit, numListItems;
float *temperature, *energy;
{       struct ListElement **CreateInitialList(), **list;
        float GetEnergy();

        list = CreateInitialList(numListItems);
        *energy = GetEnergy(list, numListItems);
        *temperature = *energy/numListItems;
        *runLimit = 100 * numListItems;
        *sucLimit = 10 * numListItems;

        return list;
}

struct ListElement **CreateInitialList(numListItems)
int numListItems;
{       int i;
        struct ListElement **list;
        void InitializeElement();

                /* Allocate memory for the array of pointers. */
        list = (struct ListElement **)malloc(sizeof(struct ListElement *) * numListItems);
        bestList = (struct ListElement **)malloc(sizeof(struct ListElement *) *
                numListItems);

                /* Create the data structures and install pointers to them into the list. */
        for(i = 0; i < numListItems; i++)
                list[i] = (struct ListElement *)malloc(sizeof(struct ListElement));

        for(i = 0; i < numListItems; i++)                /* Set initial values. */
                InitializeElement(list[i]);

        return list;
}
int num_data_pts=0, k;
double x, y, z, p, sum=0, maximum, minimum;

void InitializeElement(element)
struct ListElement *element;
{
        /* NOTE: YOU WILL HAVE TO CHANGE THIS FUNCTION. THIS
        FUNCTION SETS THE INITIAL VALUES IN THE DATA STRUCTURE.
        HERE WE JUST SET THE COORDINATES TO NUMBERS BETWEEN
ZERO
```

```
          AND 100. */
FILE *d3blk45;


          if ((d3blk45 = fopen(FILENAME,"r"))==NULL)
                    fprintf(stderr, "Cannot open the file named %s\n","d3blk45");
          while ((fscanf(d3blk45, "%1d %1d %1d %1d", &element->x, &element->y,
            &element->z, &element->p)) == 4);


          element->x = (float)rand()/RAND_MAX * 100;
          element->y = (float)rand()/RAND_MAX * 100;
          element->z = (float)rand()/RAND_MAX * 100;
          element->p = (float)rand()/RAND_MAX * 100;


float GetEnergy(list; numListItems);
struct ListElement **list;
int numListItems;
{         int i;
          float ener, energy();


          for(ener = 0, i = 0; i < numListItems - 1; i++)
                    ener += energy(list, i, i+1);


          return ener;
```

```
}


/* THIS FUNCTION CARRIES OUT THE ANNEALING PROCEDURE.  */


int Anneal(list, listSize, runLimit, temperature, sucLimit, numListItems);

struct ListElement **list;

int listSize, runLimit, sucLimit, numListItems;

float temperature;

{

        int numSuc, start, end, insertPoint, alteration, answer, i, j;

        int Oracle(), PickAlteration();

        float Ediff, EnergyDif();

        void GetSegment(), AlterList();

        FILE *bestFile;


        float GetEnergy(), ener;


        numSuc = 0;

        SucLimit = 0;


        for(i = 0; i < runLimit; i++){

                GetSegment(&start, &end, &insertPoint, numListItems);
        alteration = PickAlteration();

        Ediff = EnergyDif(list, start, end, insertPoint, numListItems, alteration);



        if(Oracle(Ediff, temperature) == YES){

            AlterList(list,start,end, insertPoint, alteration, numListItems);

            numSuc++;

            if(CHECKBEST == YES) {

                ener = GetEnergy(list,numListItems);

                if(ener < BESTENERGY) {          /* Save best list. */

                    BESTENERGY = ener;

                    for(j = 0; j < numListItems; j++) bestList[j] = list[j];

                }

            }

        }


        if(numSuc >= sucLimit) break;


    }


    return numSuc;

}
```

```
/* THIS FUNCTION DECIDES IF A NEW PATH SHOULD BE KEPT. */

int Oracle(Edif, temperature)
float Edif, temperature;
{

        if(Edif < 0.0 || exp(-Edif/temperature) > (double)rand()/RAND_MAX) return
YES;

        return NO;
}


/* THIS FUNCTION DECIDES WHETHER TO TRY REVERSING THE PATH OR
TRANSPOSING IT. */

int PickAlteration(void)
{
        if((double)rand()/RAND_MAX < 0.5) return REVERSE;

        return TRANSPOSE;
}
```

```
/* THIS FUNCTION SELECTS A SEGMENT FOR POSSIBLE ALTERATION. */

void GetSegment(start, end, insertPoint, listSize)

int *start, *end, *insertPoint, listSize;

{       int shuffle[3], done = NO, i, j, hold;


        do{             /* Select three different numbers between 0 and listSize - 1 */

                for(i = 0; i <3; i++)

                        shuffle[i] = (int)(((float)rand()/RAND_MAX)*(listSize - 1));


                if(shuffle[0] == shuffle[1] ||  /* Make sure no two are the same. */

                   shuffle[0] == shuffle[2] ||

                   shuffle[1] == shuffle[2]) ;

                else done = YES;


        }while(done == NO);



        /* Sort these numbers. */


        for(i = 0; i < 3; i++){

                for(j = i + 1; j < 3; j++){
```

```
            if(shuffle[i] > shuffle[j]){

                hold = shuffle[i];

                shuffle[i] = shuffle[j];

                shuffle[j] = hold;

            }

        }

    }


    /* Decide whether to set the insertPoint above or below the block to be altered. */


    if((double)rand()/RAND_MAX < 0.5 || shuffle[2] - 1 == shuffle[1]){

        *insertPoint = shuffle[0];

        *start = shuffle[1];

        *end = shuffle[2];

    }

    else{

        *start = shuffle[0];

        *end  = shuffle[1];

        *insertPoint = shuffle[2];

    }

}
```

```
float EnergyDif(list, start, end, insertPoint, listSize, alteration)
struct ListElement **list;
int start, end, insertPoint, alteration;
{
        float trans_cost(), reverse_cost();

        switch(alteration){
                case TRANSPOSE:
                return trans_cost(list,start,end,insertPoint,listSize);

                case REVERSE:
                return  reverse_cost(list,start,end,listSize);
        }

}

float trans_cost(list,start,end,insertPoint,listSize)
struct ListElement **list;
int start, end, insertPoint, listSize;
{       float cost, energy();

        cost = 0;
                                        /* Break the current connections. */
        if(start > 0) cost -= energy(list, start-1, start);
        if(end < listSize - 1) cost -= energy(list, end, end+1);
        if(insertPoint > 0) cost -= energy(list,insertPoint, insertPoint -1);

                                        /* Build the new connections. */

        cost += energy(list, insertPoint , end);
        if(insertPoint - 1 >= 0 ) cost += energy(list,insertPoint - 1, start);
        if(start > 0 && end < listSize - 1 )
                cost += energy(list, start - 1, end + 1);

        return cost/listSize;
}

float reverse_cost(list,start,end,listSize)
struct ListElement **list;
int start, end, listSize;
{       float cost, energy();

        cost = 0;
                                        /*Subtract the old cost. */
        if(start > 0) cost -= energy(list, start - 1, start);
        if(end < listSize- 1) cost -= energy(list,end,end+1);

                                        /* Add the new cost.  */

        if(start > 0) cost += energy(list,start - 1, end);
        if(end < listSize-1) cost += energy(list,start,end+1);

        return cost/listSize;

}

float energy(list, v1, v2)
struct ListElement **list;
int v1, v2;
{
        /* YOU MUST CHANGE THIS FUNCTION TO CALCULATE THE
        THE "ENERGY" BETWEEN TWO ELEMENTS IN A LIST.  */

        return          (float)ALEN(list[v1]->x,list[v2]->x,list[v1]->y,list[v2]->y,list[v1]-
>z,list[v2]->z,list[v1]->p,list[v2]->p);

}
```

```
void AlterList(list, start, end, insertPoint, alteration, listSize)
struct ListElement **list;
int start, end, insertPoint, alteration;
{       struct ListElement *holder, **dumList;
        int i, j;

        switch(alteration)
        {       case REVERSE:
                for(i = start, j = end; i < j; i++, j--){
                        holder = list[i];
                        list[i] = list[j];
                        list[j] = holder;
                }
                return;

                case TRANSPOSE:
                dumList = (struct ListElement **)malloc(sizeof(struct ListElment
                                *)*listSize);

                for(j = 0, i = start; i <= end; i++, j++)/* Copy group. */
                        dumList[j] = list[i];
        if(insertPoint > end){                              /* Move downward. */
          for(j=end+1, i = start; j < insertPoint; i++, j++)
                list[i] = list[j];

          for(j = insertPoint - (end - start + 1), i = 0; j < insertPoint; i++, j++)
                list[j] = dumList[i];                        /* Reinstall group. */
        }
        else {
          for(j=start - 1, i = end; j >= insertPoint; i--, j--)
                list[i] = list[j];                           /* Move upward. */

          for(j = insertPoint, i = 0; i <= (end - start); i++, j++)
                list[j] = dumList[i];

        }

        free(dumList);

        return;
    }
}
```

APPENDIX 7.1: Results of First Gold Block Model ALN Created

```
---New ALN Created, training starts over---

---Train Entry Start---

Start: TuesdayJanuary 16, 2000

Elapsed Time: days: 0, hours: 00, mins: 01, secs: 44

Train Set RMSE: 0.00351474 over 101 points

Test Set RMSE: 0.0400385 over 101 points

Tree is growable

Max Epochs: 601  Min RMSE: 0.001

Learning Rate: 0.2  Jitter is ON

LFs: 100/2601

Var 0   Epsilon: 0.0580201   Min: -1.79769e+308   Max: 1.79769e+308   WeightMin: -

1e+006  WeightMax: 1e+006

Var 1   Epsilon: 0.0001   Min: -1.79769e+308   Max: 1.79769e+308   WeightMin: -1

WeightMax: -1

90.00 % confidence interval (error) [-0.0580165, 0.0569366]
```

APPENDIX 7.2: Results of Second Gold Block Model ALN Created

```
New ALN Created, training starts over---
---Train Entry Start---
Start: Tuesday, January 16, 2000
Elapsed Time: days: 0, hours: 00, mins: 00, secs: 18
Train Set RMSE: 0.0136416 over 101 points
Test Set RMSE: 0.0396191 over 101 points
Tree is growable
Max Epochs: 601  Min RMSE: 0.001
Learning Rate: 0.2  Jitter is ON
LFs: 47/315
Var 0   Epsilon: 0.0580201   Min: -1.79769e+308   Max: 1.79769e+308   WeightMin: -
1e+006  WeightMax: 1e+006
Var 1   Epsilon: 0.028   Min: -1.79769e+308   Max: 1.79769e+308   WeightMin: -1
WeightMax: -1
90.00 % confidence interval (error) [-0.0508585, 0.0615852]
```

APPENDIX 7.3: Architecture of Second Gold Block Model ALN Created

// 3d45x1.dtr exported on January 16 16:08:40 2000

```
// ALN Decision Tree file format v1.0 (C)1994 Dendronic
Decisions Limited

VERSION = 1.0;
VARIABLES = 4;
x0 : [-1.7976931348623157e+308, 1.7976931348623157e+308];
x1 : [-1.7976931348623157e+308, 1.7976931348623157e+308];
x2 : [-1.7976931348623157e+308, 1.7976931348623157e+308];
x3 : [-1.7976931348623157e+308, 1.7976931348623157e+308];
OUTPUT = x3;
LINEARFORMS = 312;
0 : 0.0021050502052649329 (x0 - 19.023391779796487) +
0.00012814253682514747 (x1 - 26.05521542478504) +
0.00014834872015959523 (x2 - 12.728791925928162) - 1 (x3 -
0.3900000092912762);
1 : 0.0021050502052649329 (x0 - 19.023391779796487) +
0.00012814253682514747 (x1 - 26.05521542478504) +
0.00014834872015959523 (x2 - 12.728791925928162) - 1 (x3 -
0.3900000092912762);
2 : 0.0020264528204810722 (x0 - 16.387421446068561) +
0.00010753366024199396 (x1 - 27.035547575546563) +
0.0003276339416573591 (x2 - 15.262739745551485) - 1 (x3 -
0.3900000020475001);
3 : 0.0020264528204810722 (x0 - 16.387421446068561) +
0.00010753366024199396 (x1 - 27.035547575546563) +
0.0003276339416573591 (x2 - 15.262739745551485) - 1 (x3 -
0.3900000020475001);
    ....

    ....

308 : 0.0585607808868391З (x0 - 69.702793508830027) +
0.026424710735640945 (x1 - 30.187765235656741) +
0.0025320576138523486 (x2 - 16.658082333540332) - 1 (x3 -
5.7656243831892349);
309 : 0.05856078088683913 (x0 - 69.702793508830027) +
0.026424710735640945 (x1 - 30.187765235656741) +
0.0025320576138523486 (x2 - 16.658082333540332) - 1 (x3 -
5.7656243831892349);
310 : 0.090581408365491986 (x0 - 65.592153309381388) +
0.033377658600214709 (x1 - 31.052753473814786) +
0.0029184044295098369 (x2 - 12.331506506534717) - 1 (x3 -
5.7800360624512743);
```

```
311 : 0.090581408365491986 (x0 - 65.592153309381388) +
0.033377658600214709 (x1 - 31.052753473814786) +
0.0029184044295098369 (x2 - 12.331506506534717) - 1 (x3 -
5.7800360624512743);

BLOCKS = 8;
0 : MAX(MIN(97, 98, MAX(99, 100, 101), MAX(102, 103),
MAX(62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 47, 48, 49,
50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61), MAX(83, 84,
85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 73, 74, 75, 76,
77, 78, 79, 80, 81, 82, 96), MAX(121, 122, 123, 124, 125,
116, 117, 118, 109, 110, 111, 104, 105, 106, 107, 108, 112,
113, 114, 115, 119, 120, 126), MAX(147, 148, 149, 150, 151,
145, 146, 152, 129, 130, 131, 132, 133, 134, 135, 136, 127,
128, 137, 138, 139, 140, 141, 142, 143, 144, 153), MAX(168,
169, 170, 171, 172, 173, 174, 156, 157, 158, 159, 160, 161,
162, 154, 155, 163, 164, 165, 166, 167, 175, 176), MAX(17,
18, 19, 20, 21, 15, 16, 4, 5, 6, 7, 8, 2, 3, 9, 10, 11, 12,
13, 14, 0, 1, 22), MAX(37, 38, 39, 40, 41, 42, 43, 44, 45,
27, 28, 29, 30, 25, 26, 31, 32, 23, 24, 33, 34, 35, 36, 46),
MAX(184, 185, 186, 187, 179, 180, 181, 177, 178, 182, 183,
188), MAX(192, 193, 194, 195, 196, 189, 190, 191, 197, 198,
199, 200)), MIN(221, 222, 218, 219, 220, 223, 216, 217, 224,
225, 226, 208, 209, 201, 202, 203, 204, 205, 206, 207, 210,
211, 212, 213, 214, 215, 227), MIN(MAX(247, 248), 249,
MAX(MIN(MAX(250, 251, 252, 253), MAX(260, 261,
MIN(MAX(MIN(265, 266, 267), MIN(268, 269)), 270, MAX(271,
272), 273, 262, 263, 264), 274, MIN(258, 259), 256, 257,
254, 255)), 275, MIN(MAX(276, 277, 278, 279),
MAX(MIN(MAX(MIN(MAX(294, 295), 296), 297, MIN(288, 289),
290, MIN(MAX(291, 292), 293), MIN(298, 299), MIN(300, 301),
MIN(285, 286, 287), MIN(302, 303, 304)), MAX(305, 306,
307)), MIN(308, 309), MIN(282, 283, 284), MIN(280, 281)))),
MAX(310, 311), MAX(243, 244, 245), 246, 236, 237, MAX(238,
239, 240), 234, 235, MAX(241, 242), MAX(230, 231, 232,
MIN(228, 229)), 233));
…
…

7 : MAX(MIN(97, 98, MAX(99, 100, 101), MAX(102, 103),
MAX(62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 47, 48, 49,
50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61), MAX(83, 84,
85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 73, 74, 75, 76,
77, 78, 79, 80, 81, 82, 96), MAX(121, 122, 123, 124, 125,
116, 117, 118, 109, 110, 111, 104, 105, 106, 107, 108, 112,
113, 114, 115, 119, 120, 126), MAX(147, 148, 149, 150, 151,
145, 146, 152, 129, 130, 131, 132, 133, 134, 135, 136, 127,
128, 137, 138, 139, 140, 141, 142, 143, 144, 153), MAX(168,
169, 170, 171, 172, 173, 174, 156, 157, 158, 159, 160, 161,
162, 154, 155, 163, 164, 165, 166, 167, 175, 176), MAX(17,
18, 19, 20, 21, 15, 16, 4, 5, 6, 7, 8, 2, 3, 9, 10, 11, 12,
13, 14, 0, 1, 22), MAX(37, 38, 39, 40, 41, 42, 43, 44, 45,
27, 28, 29, 30, 25, 26, 31, 32, 23, 24, 33, 34, 35, 36, 46),
MAX(184, 185, 186, 187, 179, 180, 181, 177, 178, 182, 183,
188), MAX(192, 193, 194, 195, 196, 189, 190, 191, 197, 198,
199, 200)), MIN(221, 222, 218, 219, 220, 223, 216, 217, 224,
225, 226, 208, 209, 201, 202, 203, 204, 205, 206, 207, 210,
211, 212, 213, 214, 215, 227), MIN(MAX(247, 248), 249,
MAX(MIN(MAX(250, 251, 252, 253), MAX(260, 261,
MIN(MAX(MIN(265, 266, 267), MIN(268, 269)), 270, MAX(271,
272), 273, 262, 263, 264), 274, MIN(258, 259), 256, 257,
254, 255)), 275, MIN(MAX(276, 277, 278, 279),
MAX(MIN(MAX(MIN(MAX(294, 295), 296), 297, MIN(288, 289),
290, MIN(MAX(291, 292), 293), MIN(298, 299), MIN(300, 301),
MIN(285, 286, 287), MIN(302, 303, 304)), MAX(305, 306,
307)), MIN(308, 309), MIN(282, 283, 284), MIN(280, 281)))),
MAX(310, 311), MAX(243, 244, 245), 246, 236, 237, MAX(238,
239, 240), 234, 235, MAX(241, 242), MAX(230, 231, 232,
MIN(228, 229)), 233));

DTREE = 15;
0 : (x0 <= 4.2133432848335523e+306) ? 1 : 2;
1 : (x0 <= -6.5219728695327958e+306) ? 3 : 4;
2 : block 1;
3 : (x1 <= 1.1235582092889472e+308) ? 5 : 6;
4 : (x1 <= 2.9493402993834865e+307) ? 11 : 12;
5 : (x2 <= 8.4266865696671046e+306) ? 7 : 8;
6 : (x2 <= 8.4266865696671046e+306) ? 9 : 10;
7 : block 0;
8 : block 4;
9 : block 3;
10 : block 5;
11 : (x1 <= 6.6052933788275987e+306) ? 13 : 14;
12 : block 6;
13 : block 2;
14 : block 7;
```

APPENDIX 7.4: Multilayer Feed-forward Neurogenetic (Back-propagation) Network

Summary Results of NeuroGenetic Optimizer Trial run on 01-23-2001 at 20:21:29

Source Data File: 3dblk45.txt

This file contains 425 records and 4 fields.

Every 2 records were split to create

213 training records and 212 testing records.

Parameters used in this run:

Generations Run: 10

Population Size: 100

The minimum network training passes for each network were 20

The cutoff for network training passes was 50

The input neural node influence factor used was 0

The hidden neural node influence factor used was 0

The limit on hidden neurons was 64.

Selection was performed by the top 50% surviving.

Refilling of the population was done by cloning the survivors.

Mating was performed by using the TailSwap Technique.

Mutations were performed using the following technique(s):

Random Exchange technique at a rate of 25%.

Information on network rank: 1 that evolved:

Found on generation 5 after a runtime of 00:04:18

Training of this network is considered complete.

Accuracy on training set: 95.31%.

Max. accuracy on test set: 94.81%.

Accuracy on validation set: 60.00%.

***This network is a Fast-Back Propagation neural network*.**

The network employed 3 inputs and 1 hidden layer with

4 Logistic 2 Tanh 1 Linear neurons.

There were 1 output neurons using the Tanh transfer function.

APPENDIX 7.5: Generalized Regression Neurogenetic (Optimizer) Network

Best Network Report: Summary of Results of NeuroGenetic Optimizer

Trial run on 01-21-2001 at 09:15:57

Source Data File: d3blk45.txt

This file contains 425 records and 4 fields.

Every 2 records were split to create

213 training records and 212 testing records.

Parameters used in this run:

Generations Run: 50

Population Size: 100

The minimum network training passes for each network were 20

The cutoff for network training passes was 50

The input neural node influence factor used was 0

The hidden neural node influence factor used was 0

The limit on hidden neurons was 64.

Selection was performed by the top 50% surviving.

Refilling of the population was done by cloning the survivors.

Mating was performed by using the TailSwap Technique.

Mutations were performed using the following technique(s): Random Exchange technique at a rate of 25%.

Information on network rank: 1 that evolved:

Found on generation 1 after a runtime of 00:00:19

Training of this network is considered complete.

Accuracy on training set: 98.12%.

Max. accuracy on test set: 97.64%.

Accuracy on validation set: 95.00%.

This network is a Generalized Regression neural network.

The network employed 2 inputs and 2 hidden layers.

The second hidden layer used a summation transfer function.

There were 1 output neurons using a direct transfer function.

The following columns in the datafile were used:

22.5

4